

(19)日本国特許庁 (J P)

## (12) 公開特許公報 (A)

(11)特許出願公開番号

特開平6-250869

(43)公開日 平成6年(1994)9月9日

(51)Int.Cl. <sup>5</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 11/20	3 1 0 J	7313-5B		
9/46	3 4 0 D	8120-5B		
15/16	3 8 0 Z	7429-5L		
H 0 4 L 12/28				
		8732-5K	H 0 4 L 11/ 00	3 1 0 Z
		審査請求 未請求	請求項の数21	OL (全 28 頁) 最終頁に続く

(21)出願番号 特願平5-39643

(22)出願日 平成5年(1993)3月1日

(71)出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72)発明者 渡部 満

茨城県日立市大みか町七丁目1番1号 株式会社日立製作所日立研究所内

(72)発明者 山岡 弘昌

茨城県日立市大みか町五丁目2番1号 株式会社日立製作所大みか工場内

(74)代理人 弁理士 小川 勝男

(54)【発明の名称】 分散制御システム

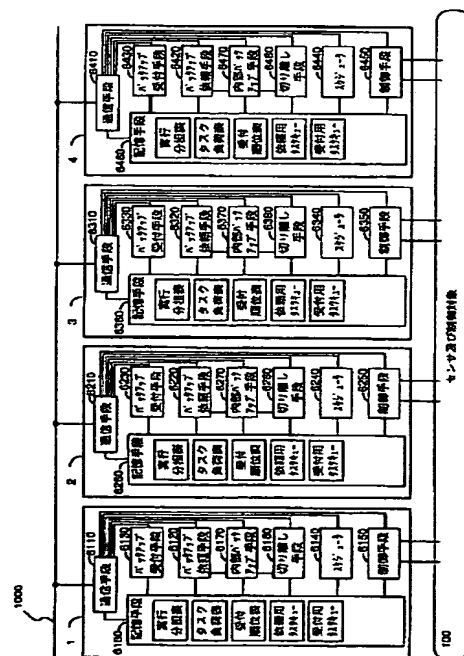
## (57)【要約】

【目的】分散制御システムにおいて、故障したコントローラあるいは負荷オーバとなったコントローラの負荷を複数のコントローラで協調して自律的に代替することを目的とする。

【構成】複数のプロセッサで構成されたコントローラを複数ネットワークで結合し、各コントローラは負荷を計測するスケジューラとコントローラ内での負荷の代替を制御する内部バックアップ手段と他のコントローラに代替を依頼するバックアップ依頼手段と他のコントローラからの代替依頼に対して自己の負荷に応じて回答するバックアップ受付手段とを備える。

【効果】自己の負荷に応じて、自コントローラ内で実行できなくなった負荷を分割して実行させることができる。また、代替できなかった負荷を他のコントローラに依頼することができる。これら、コントローラ内部の代替とコントローラ間の代替を組み合わせることによりさらに、バックアップの信頼性が高まる。

図 1



## 【特許請求の範囲】

【請求項1】複数のコントローラを分散して制御し、各コントローラは複数のタスクを実行する分散制御システムにおいて、

上記コントローラは、他のコントローラの故障又は負荷オーバーを検出する検出手段と、バックアップすべきコントローラの負荷量又はタスク量を検出し記憶する記憶手段と、上記検出手段によって故障又は負荷オーバーが検出され、故障又は負荷オーバーしたコントローラが特定されると、上記記憶手段に記憶されている上記バックアップすべきコントローラの負荷量又はタスク量に応じて、上記故障又は負荷オーバーしたコントローラの負荷又はタスクを上記バックアップすべきコントローラに分散して割当て、バックアップ制御するバックアップ手段とを有することを特徴とする分散制御システム。

【請求項2】請求項1のバックアップ手段は、上記故障又は負荷オーバーしたコントローラの負荷又はタスクを上記バックアップすべきコントローラにバックアップさせる際に、それぞれの上記バックアップすべきコントローラの予め定められた負荷量又はタスク量を越えないように上記故障したコントローラの負荷又はタスクを割り当てることを特徴とする分散制御システム。

【請求項3】請求項1のバックアップ手段は、上記故障又は負荷オーバーしたコントローラの負荷又はタスクを上記バックアップすべきコントローラにバックアップさせる際に、それぞれの上記バックアップすべきコントローラの中で、負荷量又はタスク量の小さい上記バックアップすべきコントローラから順に上記故障したコントローラの負荷又はタスクを割り当てることを特徴とする分散制御システム。

【請求項4】請求項1のバックアップ手段は、上記故障又は負荷オーバーしたコントローラの負荷又はタスクを上記バックアップすべきコントローラにバックアップさせる際に、それぞれの上記バックアップすべきコントローラの中で、負荷量又はタスク量が最も小さい上記バックアップすべきコントローラに上記故障したコントローラの負荷又はタスクを割り当てることを特徴とする分散制御システム。

【請求項5】複数のプロセッサを分散して制御し、各プロセッサは、複数のタスクを実行する分散制御コントローラにおいて、

上記プロセッサは、所定のプロセッサの故障又は負荷オーバーを検出する検出手段と、バックアップすべきプロセッサの負荷量又はタスク量を検出し記憶する記憶手段と、上記検出手段によって故障又は負荷オーバーが検出され、故障又は負荷オーバーしたプロセッサが特定されると、上記記憶手段に記憶されている上記バックアップすべきプロセッサの負荷量又はタスク量に応じて、上記故障したプロセッサの負荷又はタスクを上記バックアップすべきプロセッサに分散して割当て、バックアップ制

御するバックアップ手段とを有することを特徴とする分散制御コントローラ。

【請求項6】請求項5のバックアップ手段は、上記故障又は負荷オーバーしたプロセッサの負荷又はタスクを上記バックアップすべきプロセッサにバックアップさせる際に、それぞれの上記バックアップすべきプロセッサの予め定められた負荷量又はタスク量を越えないように上記故障又は負荷オーバーしたプロセッサの負荷又はタスクを割り当てることを特徴とする分散制御コントローラ。

【請求項7】請求項5のバックアップ手段は、上記故障又は負荷オーバーしたプロセッサの負荷又はタスクを上記バックアップすべきプロセッサにバックアップさせる際に、それぞれの上記バックアップすべきプロセッサの中で、負荷量又はタスク量の小さい上記バックアップすべきプロセッサから順に上記故障又は負荷オーバーしたプロセッサの負荷又はタスクを割り当てることを特徴とする分散制御コントローラ。

【請求項8】請求項5のバックアップ手段は、上記故障又は負荷オーバーしたプロセッサの負荷又はタスクを上記バックアップすべきプロセッサにバックアップさせる際に、それぞれの上記バックアップすべきプロセッサの中で、負荷量又はタスク量が最も小さい上記バックアップすべきプロセッサに上記故障又は負荷オーバーしたプロセッサの負荷又はタスクを割り当てることを特徴とする分散制御コントローラ。

【請求項9】請求項5のコントローラを複数有し、上記コントローラを分散して制御し、各コントローラは複数のタスクを実行する分散制御システムにおいて、

上記コントローラは、他のコントローラの故障又は負荷オーバーを検出する検出手段と、バックアップすべきコントローラの負荷量又はタスク量を検出し記憶する記憶手段と、上記検出手段によって故障又は負荷オーバーが検出され、故障又は負荷オーバーしたコントローラが特定されると、上記記憶手段に記憶されている上記バックアップすべきコントローラの負荷量又はタスク量に応じて、上記故障又は負荷オーバーしたコントローラの負荷又はタスクを上記バックアップすべきコントローラに分散して割当て、バックアップ制御するバックアップ手段とを有することを特徴とする分散制御システム。

【請求項10】ネットワークを介して制御される分散制御コントローラであって、上記分散制御コントローラの故障又は負荷オーバーを検出し、更に上記分散制御コントローラの負荷量を検出し、上記ネットワークを介して他の分散制御コントローラに検出された上記負荷を送信する検出送信手段と、上記ネットワークを介して上記分散制御コントローラの負荷を他の分散制御コントローラに代替させることを依頼する依頼手段とを有し、上記検出送信手段が上記分散制御コントローラの故障又は負荷オーバーを検出すると、その時の負荷量を上記依頼手段

によって他の分散制御コントローラに代替させることを依頼することを特徴とする分散制御コントローラ。

【請求項11】請求項10の分散制御コントローラにおいて、上記依頼手段が負荷の代替を依頼した他の分散制御コントローラからの応答を受取る受付手段とを有し、上記応答が依頼した負荷の一部分を代替したことを示したものであるなら、上記ネットワークを介して、更に他の分散制御コントローラに残りの負荷の代替を依頼することを特徴とする分散制御コントローラ。

【請求項12】請求項10又は11の分散制御コントローラにおいて、上記依頼手段は上記ネットワークを介して接続される他の分散制御コントローラの負荷量の小さい順に上記分散制御コントローラの依頼すべき負荷量を割り当てることを特徴とする分散制御コントローラ。

【請求項13】請求項10、11又は12の分散制御コントローラにおいて、上記負荷量の割当ては、上記分散制御コントローラのタスク単位で行なうことを特徴とする分散制御コントローラ。

【請求項14】ネットワークを介して他の分散制御コントローラに接続される分散制御コントローラであって、上記他の分散制御コントローラには上記他の分散制御コントローラの故障又は負荷オーバーを検出し、更に上記他の分散制御コントローラの負荷量を検出し、上記ネットワークを介して上記分散制御コントローラに検出された上記負荷を送信する検出送信手段と、上記ネットワークを介して上記他の分散制御コントローラの負荷を上記分散制御コントローラに代替させることを依頼する依頼手段とを有し、上記分散制御コントローラには上記分散制御コントローラの負荷量を検出する検出手段と、上記検出送信手段からの負荷量と上記依頼手段からの依頼を受ける受付手段と、上記検出手段によって検出された上記分散制御コントローラの負荷量と上記受付手段で受け付けた負荷量とから、バックアップすべき負荷量を定める演算手段と、上記演算手段によって演算された結果によって、依頼された負荷量のうち上記分散制御コントローラでは処理できない負荷量を上記他の分散制御コントローラに伝達する伝達手段とを有することを特徴とする分散制御コントローラ。

【請求項15】請求項14の分散制御において、上記負荷はタスクごとの単位で行なうことを特徴とする分散制御コントローラ。

【請求項16】複数の分散制御コントローラとそれぞれの上記分散制御コントローラを接続するネットワークからなる分散制御システムにおいて、それぞれの上記分散制御コントローラは故障又は負荷オーバーを検出し、更に負荷量を検出し、上記ネットワークを介して他の分散制御コントローラに検出された上記負荷を送信する検出送信手段と、上記ネットワークを介して負荷を上記他の分散制御コントローラに代替させることを依頼する依頼手段と、上記他の分散制御コントロ

ーラの検出送信手段からの負荷量と上記他の分散制御コントローラの依頼手段からの依頼を受ける受付手段と、上記検出送信手段によって検出された負荷量と上記受付手段で受け付けた負荷量とから、バックアップすべき負荷量を定める演算手段と、上記演算手段によって演算された結果によって、依頼された負荷量のうち処理できない負荷量を依頼をした上記他の分散制御コントローラに伝達する伝達手段とからなる分散制御コントローラを有することを特徴とする分散制御システム。

【請求項17】請求項16の分散制御システムにおいて、依頼された分散制御コントローラの伝達手段からの回答が負荷の一部分を代替するもの、あるいは、代替を依頼した負荷と依頼先のコントローラの負荷とを入れ替えるものであるなら、負荷を依頼された分散制御コントローラ以外の分散制御コントローラに対して、依頼を受け入れられなかった残りの負荷あるいは入れ替えられた負荷の代替を依頼することを特徴とする分散制御システム。

【請求項18】エンジン制御コントローラと、変速機制御コントローラと、サスペンション制御コントローラと、ブレーキ制御コントローラと、パワーステアリングコントローラと、故障診断制御コントローラと、ドライブレコーダコントローラと、マンマシンインターフェイス制御コントローラとから成る自動車制御システムにおいて、

請求項1、2、3、4、9、16又は17の分散制御システムからなることを特徴とする自動車制御システム。

【請求項19】エンジン制御コントローラと、変速機制御コントローラと、サスペンション制御コントローラと、ブレーキ制御コントローラと、パワーステアリングコントローラと、故障診断制御コントローラと、ドライブレコーダコントローラと、マンマシンインターフェイス制御コントローラのうち少なくとも1つのコントローラは、請求項5、6、7、8、10、11、12、13、14又は15の分散制御コントローラからなることを特徴とする自動車制御システム。

【請求項20】複数のマイクロコンピュータを用いて制御される自動車制御コントローラにおいて、上記複数のマイクロコンピュータはネットワークによって接続され、第1のマイクロコンピュータが検出した自動車の状態量を第2のマイクロコンピュータに上記ネットワークを介して伝送し、第2のマイクロコンピュータが上記状態量を用いて第2のコンピュータに接続された制御対象を制御し、第1のマイクロコンピュータが故障した場合において上記複数のマイクロコンピュータの少なくとも一つが第1のマイクロコンピュータの処理を代替し、第2のマイクロコンピュータの処理に必要な上記状態量を上記ネットワークを介して伝送し、第2のマイクロコンピュータによる制御対象の制御を継続させることを特徴とする自動車制御コントローラ。

【請求項21】複数のマイクロコンピュータからなる複数の自動車制御コントローラをネットワークを用いて制御する自動車制御システムにおいて、第1の自動車制御コントローラが検出した自動車の状態量を第2の自動車制御コントローラに上記ネットワークを介して伝送し、第2の自動車制御コントローラが上記状態量を用いて第2のコンピュータに接続された制御対象を制御し、第1の自動車制御コントローラが故障した場合において上記複数の自動車制御コントローラの少なくとも一つが第1の自動車制御コントローラの処理を代替し、第2の自動車制御コントローラの処理に必要な上記状態量を上記ネットワークを介して伝送し、第2の自動車制御コントローラによる制御対象の制御を継続させることを特徴とする自動車制御システム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は各種プロセス制御、プラント制御、車両制御等において複数の制御装置（コントローラ）を分散して制御を行なう分散制御システムに係り、特にバックアップ用のコントローラ等を用いずにそれぞれのコントローラが負荷を分担してバックアップを行なう分散制御システムに関する。

【0002】

【従来の技術】従来からプラント制御システムでは、複数のコントローラを分散して制御する分散制御システムと呼ばれる制御システムがある。この分散制御システムは複数のコントローラをネットワークで結び、発電プラントや鉄鋼圧延プラントあるいはエレベータや自動車などを制御するものである。これらの制御システムは、産業の基盤をなす重要な部分に用いられ、人命に係わる装置を制御する部分に用いられる。従って、このような分散制御システムでは稼働率と信頼性の向上が重要視される。

【0003】このような分散制御システムにおいて、1台のコントローラが故障（ダウン）した場合に他のコントローラがその負荷を代替（バックアップ）するシステムが一般的である。例えば、特開昭62-177634号と特開昭63-118860号には、あるコントローラのダウンに対してバックアップする専用のコントローラをあらかじめ用意しておくことが開示されている。さらに他の例として、特開平2-224169号と特開平3-296851号には、各計算機の負荷状況に応じて、ダウンした計算機の負荷を分割する集中管理用の計算機を有し、この集中管理用の計算機の指示に従って、複数の他の計算機でバックアップすることが開示されている。

【0004】

【発明が解決しようとする課題】ところが上記従来技術によると、次のような問題点があった。前者の技術ではバックアップ専用のコントローラを準備する必要があり、自動車制御のようにシステムのダウンサイジングが

要求されるところには不経済で、不適格であった。また、後者の技術では負荷の分散を集中管理しているためその集中管理用の計算機がダウンした場合、計算機システム全体が停止する恐れがあった。

【0005】本発明の目的は、集中管理用のコントローラやバックアップ専用のコントローラを設けずに、正常に動作しているコントローラがダウンした場合、ダウンしたコントローラの負荷を自律的に分割・分散してバックアップする分散制御システムを提供することである。

【0006】とくに、正常に動作しているコントローラの余剰性能を有効に活用し、分散制御システムの総合性能を限界まで引き上げることにある。つまり、コントローラの多重ダウンが生じた場合にも、それらのバックアップを各コントローラが協調して実施する分散制御システムを提供することにある。

【0007】

【課題を解決するための手段】上記目的を達成するため、本発明のダウンしたコントローラの負荷を自律的に分散してバックアップする分散制御システムは、他のコントローラの故障を検出する故障検出手段と、他のコントローラの負荷を検出し記憶する負荷記憶手段と、上記故障検出手段によって故障が検出され、故障したコントローラが特定されると、上記負荷記憶手段に記憶されている上記故障したコントローラ以外のコントローラの負荷に応じて、上記故障したコントローラの負荷を上記故障したコントローラ以外のコントローラに分散して割当て、バックアップ制御するバックアップ手段とを有することを特徴とする。

【0008】上記負荷記憶手段は、各コントローラが実行しているタスクとその負荷状況を示す実行分担表と、各タスクを実行したときの負荷を示すタスク負荷表とから構成され、タスク負荷表に基づいてバックアップを行なうべきコントローラの優先順位を定めることができる。また、タスク負荷表に基づいてバックアップを行なうべきコントローラの優先順位を定めた結果を格納する受付順位表を設けておくと、各コントローラの負荷状況によりどのコントローラに余裕がありバックアップを実行しやすいかの決定を高速に行なうことができる。

【0009】上記バックアップ手段は、他のコントローラにバックアップを依頼するバックアップ依頼手段と、上記受付順位表によりバックアップの受付が可能かを判定し、受付けたか否かを回答するとともに受け付けたタスクの実行を指示するバックアップ受付手段とから構成される。

【0010】また、各コントローラは複数のプロセッサで構成される。そこで、一つのコントローラ自身の耐故障性と信頼性を向上させるために、上記実行分担表には、一つのコントローラ内の各プロセッサの負荷状況とどのプロセッサに余裕があり負荷をバックアップしやすいかを示す領域を有する。それによって、バックアップ

手段は、これらの実行分担表とタスク負荷表に応じて、故障したプロセッサの負荷をバックアップするプロセッサを定める内部バックアップ手段を有している。

#### 【0011】

【作用】上記の構成により自律的にバックアップを実施するための本発明の分散制御システムの作用を、以下に説明する。

【0012】各コントローラに具備した実行分担表と、タスク負荷表と、受付順位表とにより、バックアップすべきタスクとバックアップに適するコントローラとを検索し、どのコントローラのバックアップ依頼手段を用いるかを決定することができる。この決定した結果を他のコントローラのバックアップ受付手段に通知することにより、故障したコントローラの負荷を分散することが可能となる。

【0013】また、バックアップを依頼されたコントローラのバックアップ受付手段は、自己が持つ実行分担表に格納されている自己の負荷に基づき、この通知に示されるバックアップすべき対象タスクを自己のコントローラで実行可能か否かを自ら判断し、選択的にバックアップすべき対象タスクを受付ける。この選択の結果を受付メッセージとして放送（ブロードキャスト）することにより、依頼元および他のコントローラはバックアップを実施したコントローラの負荷状況の変化を反映して、実行分担表及び受付順位表を修正する。さらに依頼先のコントローラは、受け付けられなかったバックアップすべき対象タスクを依頼元コントローラに伝達する。そこで、再び修正された受付順位表（この時は先にバックアップを実施したコントローラは負荷率が高まるため受付順位表の先頭にいないであろう）に従って、新たにバックアップすべき（受付順位表に登録された）コントローラへ未受付分を依頼する。このようにして本発明では各コントローラがシステムの実行状況を常に正確に把握することができる。また、バックアップを依頼されたコントローラがその依頼を拒否可能であるため、自律的にバックアップすべき対象タスクを分割・分散して処理することができる。

【0014】さらに、一つのコントローラに含まれる複数のプロセッサの一つが故障したときにおいても、内部バックアップ手段が実行分担表により負荷が小さいプロセッサを選択し、そのプロセッサに故障したプロセッサの負荷をすべてバックアップできるかを判定し、そのプロセッサが負荷オーバーにならないようにバックアップを実施させ、残りの負荷がある場合にはさらに他のプロセッサを選択し同様にバックアップさせる。このように内部バックアップ手段は各プロセッサが負荷オーバーにならない様に故障したプロセッサの負荷を各プロセッサの負荷限界に収まるように分割し、複数のプロセッサで次々とバックアップさせていくことができる。

#### 【0015】

【実施例】以下本発明の実施例を図面により詳細に説明する。

【0016】図1は分散制御システムのコントローラの多重ダウンに対して稼働中のコントローラが負荷を自律的に分散しバックアップする本発明の一実施例である。

【0017】プラント100のセンサ及び制御対象に接続されそれらを制御するコントローラ1, 2, 3, 4はネットワーク1000で相互に接続された分散制御システムを構成している。コントローラ1, 2, 3, 4はそれぞれ通信手段6110, 6210, 6310, 6410と、バックアップ依頼手段6120, 6220, 6320, 6420と、バックアップ受付手段6130, 6230, 6330, 6430と、スケジューラ6140, 6240, 6340, 6440と、制御手段6150, 6250, 6350, 6450と、記憶手段6160, 6260, 6360, 6460と、さらに、内部バックアップ手段6170, 6270, 6370, 6470と、切り離し手段6180, 6280, 6380, 6480とを含む各コントローラは同様の構成であるので、コントローラ1を代表例として取り上げ説明する。なお、各コントローラは後述の図3に示すとおり4個のマイクロコンピュータを持っている。さらに各マイクロコンピュータはここに示した通信手段、バックアップ依頼手段、バックアップ受付手段、内部バックアップ手段、切り離し手段、スケジューラ、制御手段の各手段を提供するに充分なハードウェアを持ち、かつ、各手段を提供するソフトウェア（タスク）を実行するに充分な処理能力を持っている。このため前記の手段は何れのマイクロコンピュータ上でも実施可能である。以下各手段の機能を説明する。

【0018】通信手段6110はネットワーク1000, バックアップ依頼手段6120, バックアップ受付手段6130, スケジューラ6140, 制御手段6150と記憶手段6160, 内部バックアップ手段6170, 切り離し手段6180に接続されており、他のコントローラとのメッセージの交換を実行し、コントローラ内の他の手段と他のコントローラの間でのメッセージあるいはデータの交換を可能にする。スケジューラ6140は自分自身に登録された複数のタスクをマイクロコンピュータに割り付ける順番及び時間間隔などを定め、かつ指示するものである。制御手段6150はプラント100のセンサ及び制御対象を制御するものである。スケジューラ6140と制御手段6150は記憶手段6160と接続されており、そこに記憶されたタスク及びデータとスケジューリング情報に基づきスケジューリングとプランと100の制御を司る。また、スケジューラ6140と制御手段6150は通信手段6110と接続されており、他のコントローラとの同期をとるために通信手段6110とネットワーク1000を介して他のコントローラとメッセージを交換する。本実施例は以上により

分散制御を実施している。

【0019】バックアップを司る手段について説明する。切り離し手段6180は内部バックアップ手段6170と通信手段6110に接続されている。この手段はコントローラ内部のハードウェアの故障あるいはソフトウェアの暴走などのダウンを検出し、ダウンしたマイクロコンピュータの動作を停止させる。また、ネットワーク1000を通して他のコントローラから自己を含むコントローラの故障を通知された場合にも同様の動作を実施する。他のコントローラからの故障の通知によってバックアップを開始できるように、切り離し手段6180は通信手段6110に接続されている。さらに、自ら故障を検出した場合および故障を通知された場合のどちらでも、どのマイクロコンピュータがダウンしたかを内部バックアップ手段6170に前記の接続を通して知らせる。

【0020】内部バックアップ手段6170は通信手段6110、記憶手段6160、切り離し手段6180とバックアップ依頼手段6120に接続されている。この手段は、切り離し手段6180からコントローラ内部のダウンを通知された場合に、図2の手順に従い、ダウンにより実行できなくなったタスクをその優先度に応じてコントローラ内でバックアップする。部分的ダウンがコントローラ全体のダウンにつながることを防ぐために、コントローラの管理運用に必要不可欠なタスクを最も優先する。ネットワーク1000の通信量のボトルネックが顕在化し通信の遅延あるいは不良が頻発することを防ぐために、他のコントローラでバックアップした場合に多量の通信を必要とするタスクが次に優先される。バックアップは負荷の軽いマイクロコンピュータから順番に実施していき、バックアップされるべきタスクがあるかぎり、次々と負荷の重いマイクロコンピュータで実施していく。途中でバックアップされるべきタスクが無くなればそこで処理を終了する。すべてのマイクロコンピュータでバックアップを実施してもバックアップされるべきタスクが残っているときはバックアップ依頼手段6120にこの状況を通知して図3のコントローラ間でのバックアップに関する手順を実行させ、本手段は実行を終了する。この図2の手順を内部バックアップ手順と呼び、後にその詳細を具体的な例により説明するが、ここでは本手段の機能を説明するうえでその概略を説明する。

【0021】まず、本手段は手順6510から手順6515にて、ダウンにより実行できなくなったタスクのバックアップを担当するマイクロコンピュータをコントローラ内から決める。この決定において記憶手段6160に格納されている各マイクロコンピュータの実行状態に関する情報を用いる。すなわち、多くのタスクをバックアップさせるために負荷の低いマイクロコンピュータあるいは余剰性能の大きいマイクロコンピュータに決定す

る。なお、これらの情報は図7に示した実行分担表と呼ぶ表に格納されている。さらに、ダウンにより実行できなくなったタスクのうちバックアップされるべきタスクを決定する。この決定では、タスクの属性に関する情報を用い、他のマイクロコンピュータでバックアップする必要のないタスクを除外する。このタスクの属性に関する情報は図8に示したタスク負荷表と呼ぶ表に格納されている。

【0022】つづいて本手段は手順6520から手順6540にて、バックアップされるべきタスクにバックアップを担当する部分がバックアップ前に実行していた複数のタスクを追加する。そしてこれらの中から、バックアップを担当するマイクロコンピュータがバックアップ後に実行すべきタスクを優先度に応じて選択する。このとき、部分的ダウンがコントローラ全体のダウンを引き起こさないようにするために、通信手段6110、バックアップ依頼手段6120、スケジューラ6140を提供するタスクなどのコントローラの管理運用に必要不可欠なタスクを最も優先して選択する。ネットワーク1000の通信量のボトルネックが顕在化して通信不良を引き起こすことがないようにするために、他のコントローラでバックアップした場合に多量の通信を必要とするタスクが次に優先して選択される。その後、その他のタスクを選択する。各優先度においてバックアップを担当するマイクロコンピュータの負荷が限界値を越えない範囲で最大負荷となる組合わせのタスクを選択する。この負荷の限界値は記憶手段6160に格納されており、限界負荷率と呼ぶ。この選択において記憶手段6160に格納されている各タスクの負荷率とこの選択における優先度とタスクの原籍に関する情報を用いる。タスクの負荷率とは着目したタスクを当該マイクロコンピュータで実行した場合の単位時間当りに必要とされる処理時間であり、あらかじめ同等の処理能力を持つマイクロコンピュータあるいは当該部分で実行したときに計測した値である。なんらかの事情で計測できないときにはあらかじめ推定し定めた値を用いる。タスクの原籍は各タスクが元々所属するコントローラあるいはその一部分を示す。なお、これらの情報は図8に示したタスク負荷表と呼ぶ表に格納されている。また、この選択において、バックアップを担当するマイクロコンピュータがバックアップ前に実行していたタスクであろうとも優先度が低い場合にはバックアップ後に実行させないことがある。このように追い出されたタスクは、バックアップされるべきタスクに追加され、他のマイクロコンピュータでバックアップされることを待つ。

【0023】つづいて本手段は手順6550にて、選択したタスクをスケジューラ6140に登録し実行手段6150にて実行を開始させる。バックアップを担当するマイクロコンピュータから追い出されたタスクはスケジューラ6140から削除しその実行を停止させる。

## 11

【0024】 つづいて本手段は手順6560から手順6590にて、バックアップされるべきタスクの状況と、コントローラ内でバックアップを担当させていないマイクロコンピュータの有無に応じて3通りの処理を実行する。まず、バックアップされるべきタスクに優先度の高いタスクがあり、かつ、コントローラ内の稼働中のマイクロコンピュータのすべてにバックアップを担当させる前述の手順を実施していない場合には、次に負荷率が低いマイクロコンピュータは、バックアップを担当するマイクロコンピュータとして選択され、再び手順6520より内部バックアップ手順を繰り返す。多くのタスクあるいは多くの負荷をコントローラ内でバックアップできるように、コントローラ内の各マイクロコンピュータに対して順番にバックアップを担当させるというこの繰り返しを設けた。一方、バックアップされるべきタスクが優先度の低いタスクのみの場合と、バックアップされるべきタスクに優先度の高いタスクがあるが、コントローラ内の稼働中のマイクロコンピュータのすべてにバックアップを担当させる前述の手順を実施した場合には、コントローラの新しい実行状況をネットワーク1000を通して放送し、バックアップされるべきタスクをコントローラ間でバックアップさせるためにバックアップ依頼手段6120にバックアップされるべきタスクが残っていることを前記接続を介して通知し図3のコントローラ間のバックアップに関する手順を開始させ、本手段の処理を終了する。さらに、一方で、バックアップされるべきタスクがない場合には、コントローラの新しい実行状況をネットワーク1000を通して放送し、本手段の処理を終了する。

【0025】 バックアップ依頼手段6120は通信手段6110と記憶手段6160と内部バックアップ手段6170とに接続されている。この手段は、内部バックアップ手段6170からバックアップされるべきタスクが残っていることを通知された場合に、図3の手順に従い、ネットワーク1000に接続された他のコントローラあるいはそのマイクロコンピュータからバックアップの依頼先を選定しバックアップを依頼する。すべてのタスクのバックアップが終了するまでこの選定と依頼を繰り返す。この手順をバックアップ依頼手順と呼び、後にその詳細を具体的な例により説明するが、ここでは本手段の機能を説明するうえでその概略を説明する。

【0026】 この手段は手順6610にて、記憶手段6160に格納されているすべてのマイクロコンピュータの負荷状態に関する情報とを用いて、バックアップを依頼するコントローラあるいはそのマイクロコンピュータを選定する。選定基準は他のコントローラのマイクロコンピュータのなかで最も負荷の小さいことである。なお、この情報は図9に示した受付順位表に格納されている。

【0027】 つづいて本手段は手順6620から手順6

## 12

640にて、さきに選定した依頼先に宛ててバックアップを依頼することを示すメッセージ6600を発信する。このメッセージ6600はバックアップされるべきタスクの情報を含む。メッセージ6600はネットワーク1000を介して送られる。そして、依頼先からの応答を待つ。応答は依頼先によってシステム全体に放送される場所の、バックアップの依頼に対する応答であることを示すメッセージ6700である。メッセージ6700はさらに依頼先がバックアップしたタスクとこのバックアップにより依頼先で実行されなくなったタスク（追い出されたタスク）を示す情報を含む。本手段はこのメッセージ6700を受信するまで待ち状態に入っており、これを受信すると待ち状態が解除されて次の手順6642に進む。

【0028】 つづいて本手段は手順6642から6648にて、依頼先がバックアップしたタスクの有無を判定し、依頼先がバックアップしたタスクがある場合にはバックアップの依頼の終了を判定する手順（手順6650から手順6670）に進む。依頼先がバックアップしたタスクがない場合には、さらに手順6644にて他のコントローラのマイクロコンピュータのすべてにバックアップを依頼したかを判定する。他のコントローラのマイクロコンピュータのすべてにバックアップを依頼していない場合には、手順6646にて他のコントローラのマイクロコンピュータの中で今回の依頼先に次いで負荷が小さいものを次回のバックアップ依頼先に指定し、手順6620に戻りバックアップの依頼を繰り返す。また、他のコントローラのマイクロコンピュータのすべてにバックアップを依頼していた場合には、バックアップが失敗している。そこでこの場合には手順6648に示すように縮退運転手段に縮退運転を開始させ、本手段はバックアップ依頼手順を終了する。

【0029】 手順6642の判定にて依頼先がバックアップしたタスクがあった場合には、つづいて、本手段は手順6660から手順6670にて、依頼先からのメッセージ6700に応じて記憶手段6160に格納されている各表を変更する。さらに、依頼先がバックアップしたタスクをバックアップされるべきタスクから削除し、依頼先から追い出されたタスクをバックアップされるべきタスクに追加する。その後、バックアップされるべきタスクの有無を判定し、バックアップされるべきタスクが有るときには手順6610に戻り依頼先の選定とバックアップの依頼を繰り返す。この依頼の繰り返しと依頼先の拒否あるいは受付により、バックアップされるべきタスクを自律的に分散してバックアップすることを可能にした。

【0030】 バックアップ受付手段6130は通信手段6110と記憶手段6160とに接続されている。この手段は、ネットワーク1000を介して他のコントローラから送られてきたバックアップを依頼するメッセージ

6600を受け取った場合に、図4の手順に従い、メッセージ6600によってバックアップを依頼されたタスクをバックアップを依頼されたマイクロコンピュータの負荷状態に応じてバックアップする。この手順ではバックアップを担当するマイクロコンピュータの負荷率が限界負荷率を越えない範囲でできる限り大きくなるように、バックアップを依頼されたタスクとそのマイクロコンピュータが実行しているタスクとの入れ替えも実施する。この入れ替えはかなり大きい負荷率のタスクをバックアップするために必要な機能である。大きな負荷率のタスクをバックアップする余裕が無くとも、小さな負荷率のタスクを追い出し余裕を作れば大きな負荷率のタスクをバックアップできることが多い。追い出されたタスクはバックアップされるべきタスクとして、さらに他のマイクロコンピュータにバックアップ依頼されそこで実行されることになる。あるいはさらにそこで入れ替えを引き起こすかも知れないが、入れ替えの度により小さな負荷率のタスクが追い出されるためバックアップできる可能性が高まってゆき、最終的にはバックアップできるであろう。この入れ替えにおいて自らのコントローラの管理運用に必要不可欠なタスクあるいは他のコントローラで実行した場合に多量の通信を必要とするタスクを追い出してしまうと、自らのコントローラのダウンやネットワーク1000の通信量のボトルネックによる通信の不良を引き起こしたりする。これを防ぐためにバックアップ後に実行するタスクとしてそれらのタスクを優先して選択するこの手順とした。また、本手段はバックアップの結果をシステム全体に知らしめる。この手順をバックアップ受付手順と呼び、後にその詳細を具体的な例により説明するが、ここでは本手段の機能を説明するうえでその概略を説明する。

【0031】この手段は手順6710から手順6730にて、バックアップされるべきタスクとバックアップを担当するマイクロコンピュータで実行されている複数のタスクの中から、バックアップを担当するマイクロコンピュータがバックアップ後に実行すべきタスクを優先度に応じて選択する。このとき、バックアップにより自らのコントローラがダウンしないようにするために、通信手段6110、バックアップ依頼手段6120、スケジューラ6140を提供するタスクなどの自らのコントローラの管理運用に必要不可欠なタスクを最も優先して選択する。ネットワーク1000の通信量のボトルネックが顕在化して通信不良を引き起こすことがないようにするために、他のコントローラで実行した場合に多量の通信を必要とするタスクが次に優先して選択される。その後、その他のタスクを選択する。各優先度においてバックアップを担当するマイクロコンピュータの負荷が限界値を越えない範囲で最大負荷となる組合わせのタスクを選択する。この負荷の限界値は記憶手段6160に格納されており、限界負荷率と呼ぶ。この選択は図2の内部

バックアップ手順の手順6530と同一である。また、この選択において、バックアップを担当するマイクロコンピュータがバックアップ前に実行していたタスクであろうとも優先度が低い場合にはバックアップ後に実行させないことがある。つづいて、本手段は選択したタスクをスケジューラ6140に登録し実行手段6150にて実行を開始させる。バックアップを担当するマイクロコンピュータから追い出されたタスクはスケジューラ6140から削除しその実行を停止させる。

10 【0032】つづいてこの手段は手順6740にて、バックアップするタスクと追い出すタスクを示す情報を含むメッセージ6700をネットワーク上に放送（ブロードキャスト）する。このメッセージ6700により自らのコントローラの実行状態の変化を他のコントローラのすべてに知らしめる。さらに、本手段はすべてのコントローラは各々の記憶手段6160、6260、6360、6460に格納されている実行分担表とタスク負荷表と受付順位表をこのメッセージ6700によって書き換える。また前述したように、このメッセージ6700

20 に基づいて依頼元のバックアップ依頼手段（6220、6320、6420のいずれか）はバックアップされるべきタスクからバックアップされたタスクを削除し、追い出されたタスクをバックアップされるべきタスクに追加する。

【0033】このように1台のダウンしたコントローラが実行していた複数のタスクがバックアップを依頼された稼働中のコントローラの負荷に応じて分割されることにより、最終的には複数のコントローラでバックアップされる。つまり、本発明の分散制御システムは1台のコントローラのダウンを複数台の稼働中のコントローラが自らの負荷状況に応じて分担してバックアップするものである。さらに、このバックアップにおいてネットワークの通信量増加を抑えるために、タスクの通信量に関する属性に応じて分担してバックアップするものである。

【0034】この分散制御システムの物理的な構成を図5に示す。コントローラ1、2、3、4は同一の構成であり、各々がネットワーク1000と接続されたネットワーク送受信部（コネクタ）41a、41b、41c、41dと、これらのネットワーク送受信部（コネクタ）の各々とシリアルデータ用のバス71a、71b、71c、71dを介して接続された4個ずつのマイクロコンピュータ（MC1、MC2、MC3、MC4）701～704、711～714、721～724、731～734と、これらのマイクロコンピュータ（MC1、MC2、MC3、MC4）とバス70a、70b、70c、70dを介して接続されたグローバルメモリ82a、82b、82c、82dと、これらのマイクロコンピュータ（MC1、MC2、MC3、MC4）と接続された入出力処理装置（I/O）83a、83b、83c、83dと、これらのマイクロコンピュータに接続され故障の

40

50



通知を行うバス70bを含む。各コントローラにはDC1, DC2, DC3, DC4という名前を付けた。各コントローラ内の各マイクロコンピュータにはMC1, MC2, MC3, MC4という名前を付けた。ここで、各マイクロコンピュータはその中に記したタスクを実行しているものとした。ここでタスクNC1, NC2, NC3, NC4はネットワーク1000を介したコントローラ間のメッセージの送受信を制御するタスクであり、図1の通信手段6110, 6210, 6310, 6410を提供する。また、タスクCM1, CM2, CM3, CM4は各コントローラ1, 2, 3, 4でのタスクの実行を制御するタスクであり、図1のバックアップ依頼手段6120, 6220, 6320, 6420と、バックアップ受付手段6130, 6230, 6330, 6430を提供する。頭文字がTで始まるタスクは各コントローラが実行する制御演算などであり、図1の制御手段を提供する。OS11などのOSで始まるタスクはバックアップ関係の機能を組み込んだリアルタイムオペレーティングシステムであり、各マイクロコンピュータで実行されている。これら各々が図1の内部バックアップ手段6170, 6270, 6370, 6470を提供する。またこれらのリアルタイムオペレーティングシステムの各々が独立に各々のマイクロコンピュータに関するスケジューリングを実施している。すなわち、各々が各コントローラのスケジューラ6140, 6240, 6340, 6440を各マイクロコンピュータに関する部分に分割して提供している。

【0035】各マイクロコンピュータは同一の構成であり、その構成をマイクロコンピュータ(MC1)701を例に取り上げ図4を用いて説明する。マイクロコンピュータ(MC1)701はタスクを実行するプロセッサ(PR)6811と、タスクの中に含まれたニューロ演算を加速するためのニューラルエンジン(NE)6812と、タスクの中に含まれたシーケンス制御用の演算を加速するシーケンスエンジン(SE)6813と、グローバルメモリ(GM)82aに対する他のマイクロコンピュータとのアクセス競合による性能低下を防止する為にグローバルメモリ(GM)82aの内容の一部分を複製して格納したローカルメモリ(LM)6816と、アクセスすべきメモリがグローバルメモリ(GM)82aのときバス70aとマイクロコンピュータ内のバス6819を結合するバスインターフェース(BIF)6817と、ローカルメモリ(LM)6816とグローバルメモリ(GM)82aの間で高速な転送を実行するダイレクトメモリアクセス制御(DMAC)6818と、スケジューリングのための時計であるフリーランタイム(FRT)6815と、ネットワーク送受信部41aと接続されシリアルデータの並列データ化あるいはその逆の操作と符号化/復号化などを実行するランアダプタ(LA)6814と、各構成要素の故障を検出してマイクロ

コンピュータの動作を停止させるとともに故障を他のマイクロコンピュータにバス70bを通して通知する故障制御回路6820を含む。さらに、各構成要素はバス6819で接続されている。

【0036】ここで、故障制御回路6820は他のマイクロコンピュータの故障制御回路と協同してバス70bを通信路として用いることにより切り離し手段を提供する。例えば図3のマイクロコンピュータ702が故障した場合には、マイクロコンピュータ702の故障制御回路が故障を検出し、他のマイクロコンピュータ701, 703, 704に対してバス70bを通してマイクロコンピュータ702の故障を通知し、同時にマイクロコンピュータ702のプロセッサ(PR)の動作を停止させる。さらに、マイクロコンピュータ701, 703, 704の各故障制御回路は各々のプロセッサPRで実行されている内部バックアップ手段を提供するタスクOS11, OS13, OS14に内部バックアップ手順を開始させる。このように図1の切り離し手段6180における機能は物理的には4つのマイクロコンピュータに分担されている。

【0037】さらに、図3の各コントローラのなかのすべてのマイクロコンピュータがランアダプタ(LA)を持つため、いずれのマイクロコンピュータも通信手段6110, 6210, 6310, 6410として動作可能である。従来技術では通信関係の手段が冗長化されておらず信頼性が不十分であったが本実施例の構成により冗長化され信頼性が向上する。各コントローラ内の4つのランアダプタ(LA)のなかで、実際に通信手段6110, 6210, 6310, 6410となるものはタスクNC1, NC2, NC3, NC4が実行されているマイクロコンピュータのランアダプタ(LA)である。つまり通信手段として動作中のランアダプタ(LA)が故障して通信が不通になっても、タスクNC1, NC2, NC3, NC4をコントローラ内でバックアップすることで通信を回復することができる。また、従来技術ではプロセッサあるいはマイクロコンピュータがローカルメモリを持った場合、そのプロセッサあるいはマイクロコンピュータがダウンすると、ローカルメモリに蓄えられた制御用の学習結果のデータなどにアクセスできないという問題があった。そこでローカルメモリ(LM)6816をデュアルポートメモリで構成し、バス6819のほかにバス70aにも接続した。この接続により他のマイクロコンピュータからバス71aを介してローカルメモリ(LM)6816へアクセスできるようにした。コントローラ内の4つのマイクロコンピュータ(MC1, MC2, MC3, MC4)のローカルメモリ(LM)とグローバルメモリ(GM)によって図1の記憶手段6160, 6260, 6360, 6460を構成している。

【0038】以下では本実施例のバックアップ方法について詳細に例示する。図7は図3の動作状態における実

17

行分担表であり、各記憶手段6160、6260、6360、6460に格納されている。この表の内容はマイクロコンピュータを特定可能な名前と、そのマイクロコンピュータが実行しているタスクの名称と、それらのタスクによってマイクロコンピュータが被る負荷率と、各コントローラ内での各マイクロコンピュータに負荷の軽い順に1番から番号を付けた順位（内部負荷順位）である。本例ではコントローラ名とマイクロコンピュータ名を連結しマイクロコンピュータを特定可能な名称としている。このほかの特定する方式として各々異なる番号を与えるものがある。このマイクロコンピュータを特定するために用いる方式は本発明の主旨とは関係なくどのような方式でもよい。本表を用いて内部バックアップ手段6170、6270、6370、6470は、バックアップされるべきタスクをダウンしたマイクロコンピュータ名から検索し、コントローラ内で負荷の小さいマイクロコンピュータから大きいマイクロコンピュータへ順番にバックアップを担当させるために内部負荷順位を検索する。また、バックアップ受付手段6130、6230、6330、6440がバックアップを依頼されたマイクロコンピュータ名からそのマイクロコンピュータが実行しているタスク名を検索するために実行分担表を用いる。なお、内部負荷順位が無くても、内部バックアップ手段はマイクロコンピュータ名と負荷率をもとに内部負荷順位と同等の情報を生成し上記の処理を実行可能であるが、バックアップの途中でこの情報を生成するためバックアップを終了するまでの時間が長くなる。バックアップを終了するまでの時間を短縮するために、本実施例では通常の稼働状態の時に内部負荷順位を生成しこの実行分担表に記憶しておく方式にした。

【0039】図8は図5の動作状態におけるタスク負荷表であり、各記憶手段6160、6260、6360、6460に格納されている。この表の内容はタスクの名前と、そのタスクを実行したときの負荷率と、バックアップに関係する属性と、タスクがどのマイクロコンピュータのものかを示す原籍である。属性は4つあり、原籍のマイクロコンピュータにおいて必要不可欠であるが他のマイクロコンピュータでバックアップする必要のない（あるいはしてはならない）タスクであることを示す“fixed”と、コントローラの管理運用に必要なタスクであることを示す“internally”と、通信量が多いことを示す“communicative”と、原籍以外のコントローラで実行してもよいタスクであることを示す“somewhere”である。よって、他のマイクロコンピュータでバックアップされるべきタスクは属性が“fixed”でないものである。これらのあるマイクロコンピュータにおける優先度として扱い優先度が高いものから降順に並べると、最優先のものはこのマイクロコンピュータが原籍である“fixed”、次に優先されるものはこのマイクロコンピュータが原籍である“internally”、さらに次に優先されるもの

18

はこのマイクロコンピュータが原籍である“communicative”、優先度の最も低いものはこのマイクロコンピュータが原籍である“somewhere”とこのマイクロコンピュータが原籍でない“fixed”、“internally”、“communicative”と“somewhere”である。ただし、原籍以外のマイクロコンピュータに属性が“fixed”や“internally”のタスクをバックアップさせることはコントローラあるいはシステムの不具合を生じるため、バックアップ手順の中でこれを防いでいる。すなわち、原籍のマイクロコンピュータあるいはそれを含むコントローラ（原籍のコントローラと呼ぶ）にて最優先でバックアップされるようにしている。内部バックアップ手段6170、6270、6370、6470と、バックアップ受付手段6130、6230、6330、6440がバックアップされるべきタスクとバックアップを担当するマイクロコンピュータが実行していたタスクとから各々のタスクの優先度と原籍と負荷率を基準としてバックアップ後に実行すべきタスクを選択する時に、タスク名からそれらの内容を知るためにこのタスク負荷表を用いる。

【0040】図9は図5の動作状態における受付順位表であり、各記憶手段6160、6260、6360、6460に格納されている。本表はバックアップ依頼手段6120、6220、6320、6420が他のコントローラのマイクロコンピュータの中から、最小負荷率のマイクロコンピュータの名称を最初に検索し、つづいて必要ならば順番に負荷の大きいものの名称を検索するために用いる。この表はマイクロコンピュータ名を負荷率をキーとして昇順に並べたものである。よって、実行分担表に本表の情報も含まれているが、実行分担表で最も負荷率の小さいマイクロコンピュータを検索するためには相当な時間がかかる。そこで、検索時間短縮を目的として本表を設けた。本表では先頭のマイクロコンピュータ名を読みだすだけですべてのマイクロコンピュータの中で最小負荷率のマイクロコンピュータ名を知ることができる。それがバックアップを依頼するコントローラに属するときは次の順位を読みだす。この繰り返しで他のコントローラのマイクロコンピュータで最も負荷が小さいものを選択できる。本表はダウンやバックアップの度に書き換えるが、一度並べ帰られているので削除、2分検索、挿入の3ステップでこの表の昇順を維持する。このため、この受付順位表を用いた場合にバックアップ時に実行分担表の内容をソーティングを実施する必要がなく処理が高速になる。すなわち、バックアップに要する時間が短縮できる。

【0041】ここで、実行分担表とタスク負荷表と受付順位表の内容の関係を説明する。マイクロコンピュータDC1、MC1が実行しているタスクは実行分担表よりタスクOS11、NC1、CM1、T11である。このときの負荷率はタスク負荷表からタスクOS11が5%、NC1が26%、CM1が28%、T11が13%

であるからこれらの和である72%となる。実行分担表の例示で省略されたマイクロコンピュータの負荷率はこの値より充分大きいとして、マイクロコンピュータDC1. MC1の負荷率は昇順で3番目であるから、受付順位表では順位が3番目になっている。

【0042】ここで、本実施例のバックアップ手順の理解の助けとして、図10に示すようにコントローラ1のマイクロコンピュータ(DC1. MC1)701がダウンした場合において、タスクNC1, CM1, T11が分割してバックアップされる過程とそれにより追い出されたタスクがさらにまた他のマイクロコンピュータによってバックアップされる過程を説明する。ここで前述の限界負荷率は90%に定められているものとする。

【0043】(FD0)コントローラ1のマイクロコンピュータ701のプロセッサ6811が故障したものとする。この故障により通信手段6110とバックアップ依頼手段6120とバックアップ受付手段6130がダウンし、スケジューラ6140と制御手段6150の各一部分がダウンした。

【0044】(FD1)コントローラ1の切り離し手段6180がマイクロコンピュータ701のダウンを検出すると、このマイクロコンピュータを停止させる。同時に内部バックアップ手段6170にマイクロコンピュータ701のダウンを通知し、内部バックアップ手段6170に図4の内部バックアップ手順を開始させる。より詳細には、マイクロコンピュータ701の故障制御回路(FD)6820がプロセッサ(PR)6811の故障を検出すると、バス70bを通して他のマイクロコンピュータの故障制御回路(FD)にマイクロコンピュータ701の故障を通知する。その後マイクロコンピュータ701を停止させる。マイクロコンピュータ702, 703, 704の故障制御回路(FD)6820は故障の通知を受けると各々のプロセッサ6811に対してバス70aの割り込み要求信号線を用いて割り込み処理を要求する。この割り込み処理が、内部バックアップ手段6170を提供するタスクOS12, OS13, OS14に内部バックアップ手順を開始させる。なお、停止されたマイクロコンピュータのローカルメモリ(LM)6816はバス70aと接続されているため、このマイクロコンピュータが停止していても他のマイクロコンピュータからアクセス可能である。この接続はタスクをバックアップし再開するときに必要なデータをマイクロコンピュータが停止していても取得できるようにしたものである。

【0045】以下、内部バックアップ手段6170が実行する図4に示した内部バックアップ手順について説明する。

【0046】(OS0)内部バックアップ手段6170が起動される。詳細には稼働中のマイクロコンピュータのすべてにおいて内部バックアップを司るタスクが各々独立に図4の内部バックアップ手順を実行する。図10

の例示ではマイクロコンピュータ(DC1. MC2)702でタスクOS12が内部バックアップ手順を実行する。マイクロコンピュータ(DC1. MC3)703でタスクOS13が内部バックアップ手順を実行する。マイクロコンピュータ(DC1. MC4)704でタスクOS14が内部バックアップ手順を実行する。バックアップを担当しないタスクが自ら停止し、自動的にバックアップを実施するものが決定する。マイクロコンピュータの間にバックアップに関する特定の依存関係がないためコントローラ内を均一のハードウェア及び均一のソフトウェア体形とすることが可能となり、コントローラの構築が容易になる。このように図4の内部バックアップ手順を構築した。次のステップでこの決定方法を説明する。

【0047】(OS1)手順6510:内部バックアップ手段6170は上記の稼働中のマイクロコンピュータのなかで負荷が最も小さいマイクロコンピュータをバックアップを実施するマイクロコンピュータに決定する。

【0048】具体的には前記のタスクの各々が実行分担表からダウンしたマイクロコンピュータの内部負荷順位と自らの内部負荷順位を読みだす。ダウンしたマイクロコンピュータの負荷が最も小さいのならば内部負荷順位は前述のように1番である。よって、その内部負荷順位が1ならば稼働中のマイクロコンピュータの中で負荷が最も小さいものは内部負荷順位が2番目である。この場合には、自らの内部負荷順位が2番であることを判定することにより、自らがバックアップを担当するか否かを定める。内部負荷順位が2番ならばバックアップを担当することになり、手順6520に進む。もし、内部負荷順位が2番でなければ内部バックアップ手順を終了し、バックアップを担当しない。一方、ダウンしたマイクロコンピュータの負荷より稼働中のマイクロコンピュータの負荷が小さい場合には、稼働中のマイクロコンピュータの中で負荷が最も小さいものは内部負荷順位が1番目である。この場合には、自らの内部負荷順位が1番であることを判定することにより、自らがバックアップを担当するか否かを定める。内部負荷順位が1番ならばバックアップを担当することになり、手順6520に進む。もし、内部負荷順位が1番でなければ内部バックアップ手順を終了し、バックアップを担当しない。図10の例示ではダウンしたマイクロコンピュータ(DC1. MC1)701の内部負荷順位が図7の実行分担表に示すとおり2番であり、マイクロコンピュータ(DC1. MC2)702の内部負荷順位が4番であるため、タスクOS12は手順6510で内部バックアップ手順を終了する。また、マイクロコンピュータ(DC1. MC3)703の内部負荷順位が3番であるため、タスクOS13は手順6510で内部バックアップ手順を終了する。一方、マイクロコンピュータ(DC1. MC4)704の内部負荷順位は1番であるため、タスクOS14は手順

6510から手順6520に進み、内部バックアップ手順を継続する。よって、マイクロコンピュータ（DC1、MC4）704がまず初めにバックアップを実施する。

【0049】（OS2）手順6515：内部バックアップ手段6170はダウンにより実行できなくなったタスクのうちバックアップする必要の有るものをタスクキューに登録する。詳細にはまず受付用タスクキューを初期化する。そして、実行分担表からダウンしたマイクロコンピュータの名前に対応するタスク名を読みだし、タスク負荷表からそれらのタスクに対する負荷率と属性を読み出す。そして、属性がバックアップする必要がないこと、あるいは、バックアップしてはならないことを示す“fixed”のものを除外して、バックアップされるべきタスクとして受付用タスクキューに登録する。図10の例示ではダウンしたマイクロコンピュータ（DC1、MC1）701の名前DC1、MC1に対応するタスク名であるOS11とNC1とCM1とT11が実行分担表から読みだされ、タスク負荷表から各々の負荷率5%、26%、28%、13%と属性“fixed”，“communicative”，“somewhere”が読みだされる。そして、タスクの属性が“fixed”であるタスクOS11に関する情報を除いて、これらの内容が受付用タスクキューに登録される。

【0050】（OS3）手順6520：内部バックアップ手段6170はバックアップを担当するマイクロコンピュータが実行しているタスクを受付用タスクキューに追加する。詳細には実行分担表からバックアップを担当するマイクロコンピュータの名前に対応するタスク名を読みだし、タスク負荷表からそれらのタスクに対する負荷率と属性を読み出す。そしてそれらのタスクを受付用タスクキューに追加する。同時にそれらの負荷率及び属性も追加する。

【0051】この手順以降はコントローラ内部のすべてのマイクロコンピュータがバックアップを担当するまで、あるいは、バックアップされるべきタスクがなくなるまで繰り返し実行される。図10の例示における今回の実行では、バックアップを担当するマイクロコンピュータ704の名前DC1、MC4に対応するタスク名であるOS14とT17とT18が実行分担表から読みだされ、タスク負荷表から各々の負荷率5%、25%、26%と属性“fixed”，“internally”，“internally”，“communicative”が読みだされる。これらが受付用タスクキューに追加される。

【0052】（OS4）手順6530：内部バックアップ手段6170はバックアップを担当するマイクロコンピュータがバックアップ後に実行すべきタスクを受付用タスクキューから選択する。具体的には、受付用タスクキューをタスクの属性による優先順位と負荷率で並べ変えて、優先順位の高いタスクのなかで限界負荷率に最も近くなる組合わせを選択し、さらにすべてを選択しても

負荷率に余裕があるならば次の優先度のタスクのなかでその余裕に最も近くなる組合わせを選択する。何れかの優先順位のタスクの中で選択できないものが生じるまで、この組合わせと選択を繰り返す。図10の例示では受付用タスクキューが図11に示した内容になり、タスクOS14とNC1とCM1とT17が最終的に選択される。ここで、バックアップされるタスクはNC1とCM1であり、バックアップされないタスクはT11であり、追い出されるタスクはT18である。

10 【0053】（OS5）手順6540：前手順により選択されたタスクを受付用タスクキューから削除する。具体的には、受付用タスクキューから前手順で選択されたタスクの名称とその負荷率と属性を削除する。図10の例示ではタスクOS14とNC1とCM1とT17が最終的に選択された。当手順では、これらのタスクを受付用タスクキューから削除する。受付用タスクキューに残されたタスクは、バックアップされなかったタスクT11と追い出されたタスクT18である。

20 【0054】（OS6）手順6550：手順6530にて選択されたタスクをスケジューラ6140に登録し実行を開始させる。具体的には、バックアップを担当するマイクロコンピュータのスケジューリングを担当するタスクに対してバックアップされるタスクを追加登録し、追い出されるタスクの登録を削除する。図10の例示ではバックアップを担当するマイクロコンピュータ704のスケジューリングを担当するタスク、すなわち、リアルタイムオペレーティングシステムであるタスクOS14に、バックアップされるタスクであるタスクNC1とCM1を追加登録し実行を開始させる。また、追い出されるタスクであるタスクT18をスケジューラから削除し実行を停止させる。ここにおいて、マイクロコンピュータ701の故障によりダウンしていた通信手段6110とバックアップ依頼手段6120とバックアップ受付手段6130が回復された。以降、コントローラ間のバックアップや制御タスクの実行に必要なネットワーク1000との通信が実行できる。

30 【0055】（OS7）手順6560：当手順はバックアップの状況に応じて内部バックアップ手順を3通りに分岐させる。一つは完全にバックアップが終了したのでバックアップに関する手順を終了させる処理への分岐である。二つは内部でのバックアップを完了しコントローラ間におけるバックアップを開始させる処理への分岐である。三つは内部でのバックアップを繰り返す処理への分岐である。手順6560の中でこれらの分岐を手順6562、6564、6566が実施している。手順6562はバックアップすべきタスク、すなわち、受付用タスクキューに残っているタスクの中で優先度の高いタスクが有るか否かを判定し、有るならば処理は手順6564へ進む。無いならば処理は手順6566へ進む。手順6564はコントローラ内のすべてのマイクロコンピュータ

にバックアップを担当させたか否かを判定し、すべてに担当させた後ならば処理は手順6574以降のコントローラ間におけるバックアップを開始させる処理へ分岐する。すべてに担当させていないならば処理は手順6580以降の内部でのバックアップを繰り返す処理へ分岐する。手順6566はバックアップすべきタスクが残っているか否かを、すなわち、受付用タスクキューにタスクが残っているか否かを判定し、タスクが残っているならば処理は手順6574以降のコントローラ間におけるバックアップを開始させる処理へ分岐する。タスクが残っていないならば処理は手順6572以降のバックアップに関する手順を終了させる処理へ分岐する。

【0056】図10の例示において、今回は、属性が“communicative”であり優先度が高いタスクT11が受付タスクキューに残っており、尚且つ、まだマイクロコンピュータ704にバックアップを担当させたのみであるため、手順6580以降の内部でのバックアップを繰り返す処理へ分岐する。

【0057】(OS8) 手順6580：今回バックアップを担当したマイクロコンピュータの次に負荷が小さいものを次回のバックアップを担当するマイクロコンピュータに選定し、次回のバックアップを実施するために手順6520へ戻る。詳細には今回バックアップを担当したマイクロコンピュータの内部受付順位とダウンしたマイクロコンピュータの内部受付順位を実行分担表から読み出す。前者の値に1を加え次の内部受付順位を求める。この値が後者の受付順位であった場合にはさらに1を加えてダウンしたマイクロコンピュータを指定してしまうことを防ぐ。コントローラ内で内部受付順位がこの値であるマイクロコンピュータを実行分担表から求め、求めたマイクロコンピュータを次回にバックアップを担当するものとして選択する。図10の例示では今回バックアップを担当したマイクロコンピュータ704の内部受付順位が1番であり、ダウンしたマイクロコンピュータの内部受付順位が2番であるため、次回にバックアップを担当するマイクロコンピュータの内部受付順位は3番になる。よって実行分担表においてコントローラ1内で内部受付順位が3番であるマイクロコンピュータ703が次回のバックアップを担当するものとして選択される。ここから2回目の内部バックアップにはいる。すでに説明した各手順は図10の例示に関する説明のみを記述する。

【0058】(OS9) 手順6520：図10の例示における今回の実行では、バックアップをマイクロコンピュータ703が担当する。そこで、その名前DC1、MC3に対応するタスク名であるOS13とT15とT16が実行分担表から読みだされ、タスク負荷表から各々の負荷率5%、28%、41%と属性“fixed”、“somewhere”、“communicative”が読みだされる。これらが受付用タスクキューに追加される。

【0059】(OS10) 手順6530：図10の例示では受付用タスクキューが図12に示した内容になり、タスクOS13とT15とT16とT11が最終的に選択される。ここで、バックアップされるタスクはT11であり、バックアップされないタスクはT18であり、追いつ出されるタスクは無い。

【0060】(OS11) 手順6540：図10の例示ではタスクOS13とT15とT16とT11が最終的に選択された。当手順では、これらのタスクを受付用タスクキューから削除する。受付用タスクキューに残されたタスクは、バックアップされなかったタスクT18のみである。

【0061】(OS12) 手順6550：図10の例示ではバックアップを担当するマイクロコンピュータ703のスケジューリングを担当するタスク、すなわち、リアルタイムオペレーティングシステムであるタスクOS13に、バックアップされるタスクであるタスクT11を追加登録し実行を開始させる。また、追いつ出されるタスクが無いため、スケジューラから削除され実行を停止させられるものは無い。

【0062】(OS13) 手順6560：図10の例示において、今回は、属性が“somewhere”であり優先する必要がないタスクT18のみが受付タスクキューに残っている。すなわち、この状態は優先度の高いタスクは残っていないが、バックアップすべきタスクが残っている状態である。よって、当手順は処理を手順6574以降のコントローラ間におけるバックアップを開始させる処理へ分岐させる。

【0063】(OS14) 手順6574：新しい実行状態を示すメッセージ6700をネットワーク1000に放送（ブロードキャスト）する。このメッセージにより他のコントローラはマイクロコンピュータのダウンと、そのバックアップを担当したマイクロコンピュータの新しい実行状態を知ることができる。このメッセージ6700の内容に合わせて、各コントローラは各々の記憶手段に格納している実行分担表とタスク負荷表と受付順位表を修正する。図10の例示において、メッセージ6700はダウン情報としてマイクロコンピュータ701のダウンを示す。さらに、新しい実行状況としてマイクロコンピュータ703が実行しているタスクOS13、T15、T16、T11と、マイクロコンピュータ704が実行しているタスクOS14、NC1、CM1、T17とを示す。修正された実行分担表と受付順位表を図14、図15に示す。

【0064】(OS15) 手順6590：コントローラ間でのバックアップを実施するために、バックアップ依頼手段6120を起動する。詳細には残ったバックアップされるべきタスクを受付用タスクキューからバックアップ依頼手段6120が用いる依頼用タスクキューに移し、受付用タスクキューを空にする。そして、バックア

バックアップ手段6120にバックアップされるべきタスクが残っていることを通知する。この通知はリアルタイムオペレーティングシステムのタスク間通信機能を利用して行われてもよい。あるいは、ネットワーク1000を介した自コントローラから自コントローラ宛のメッセージ通信を利用して行われてもよい。

【0065】(OS16)以上にて内部バックアップ手順を終了する。

【0066】続いて、コントローラ間でバックアップ手順に移行し、タスクT18を他のコントローラでバックアップする。

【0067】(CM1)コントローラ1のバックアップ依頼手段6120が起動され、図3のバックアップ依頼手順の実行を開始する。

【0068】(CM2)手順6610:バックアップ依頼手段6120が受付順位表を用い、ダウンしたマイクロコンピュータを吹く不コントローラとは異なる他のコントローラのマイクロコンピュータの中で負荷が最も小さいものをバックアップの依頼先に決定する。詳細には受付順位表から順位が1番のマイクロコンピュータ名を読みだす。そのマイクロコンピュータが他のコントローラに所属するならばそこで読みだしを停止し、そのマイクロコンピュータをバックアップの依頼先にする。もし、ダウンしたマイクロコンピュータと同一のコントローラに所属しているならば、再び受付順位表から次の順位のマイクロコンピュータ名を読みだす。そして、他のコントローラに所属しているか否かを判定する。他のコントローラに所属するマイクロコンピュータが読みだされるまで以上を繰り返す。図10の例示ではダウンしたマイクロコンピュータを含むコントローラ内でのバックアップを終了しているため、図9の受付順位表は図15に変更されている。この図15の受付順位表から順位1番のマイクロコンピュータ名を読みだす。この名前はDC2、MC2であり、このマイクロコンピュータはコントローラDC2に属する。ダウンしたマイクロコンピュータを含むコントローラはDC1であるため、読みだされたマイクロコンピュータはダウンしたマイクロコンピュータを含むコントローラとは異なったコントローラに属する。よって次の順位のマイクロコンピュータ名を読みだす処理に戻らずに、このマイクロコンピュータ(DC2、MC2)712をバックアップを依頼するマイクロコンピュータに決定する。このマイクロコンピュータを依頼先と呼ぶ。

【0069】(CM3)手順6620:バックアップ依頼手段6120がバックアップの依頼であることを示す情報と、依頼用タスクキューに登録されているすべてのタスク名を示す情報とを含むメッセージ6600を作成し、依頼先に宛てて発信する。このメッセージを依頼メッセージと呼ぶ。図10の例示では依頼メッセージ6600はタスクT18のバックアップを依頼するものであ

り、依頼先であるマイクロコンピュータ712に宛ててネットワーク1000を介して送付される。(CM4)手順6630:バックアップ依頼手段6120はバックアップを依頼したメッセージ6600に対する回答としてバックアップの依頼先から放送されるメッセージ6700を待つ。

【0070】以後、メッセージ6700が放送されるまでバックアップ依頼手段6120はバックアップ依頼手順を中断して、待ち状態となる。ここからは依頼先のバックアップ受付手段6230の動作について説明する。

【0071】(CM5)依頼先のマイクロコンピュータ712を含むコントローラ2に於て、メッセージ6500を通信手段6210が受け取り、バックアップ受付手段6230を起動する。バックアップ受付手段6230は図4のバックアップ受付手順を開始する。ここで、バックアップ受付手段6230はそれ自身を含むコントローラの記憶手段6260に格納されている受付用タスクキューとタスク負荷表を用いる。

【0072】(CM6)手順6710:バックアップ受付手段6130がメッセージ6600によって依頼されたタスクとバックアップを依頼されたマイクロコンピュータが実行しているタスクを受付用タスクキューに登録する。詳細にはバックアップ受付手段6230はまず記憶手段6260内の受付用タスクキューを初期化する。続いて、メッセージ6600から依頼されたタスクの名称をとりだし、記憶手段6260内のタスク負荷表からそのタスク名に対応する負荷率と属性と原籍を読みだし、先に初期化した受付用タスクキューに登録する。さらに、記憶手段6260内の実行分担表から依頼先のマイクロコンピュータの名前に対応するタスク名を読みだし、タスク負荷表からそれらのタスクに対する負荷率と属性を読みだす。そしてそれらのタスクを受付用タスクキューに追加する。同時にそれらの負荷率及び属性も追加する。

【0073】図10の例示では、メッセージ6600によりバックアップを依頼されたタスクの名称T18を受付用タスクキューに登録する。同時に、タスク負荷表からその名称に対応した負荷率26%と属性“somewhere”と原籍DC1、MC4を読みだしこれらに登録する。さらに、バックアップを依頼されたマイクロコンピュータ712が実行しているタスクの名称として、そのマイクロコンピュータ名DC2、MC2に対応するタスク名OS22とT21とT22を実行分担表から読みだす。さらに、タスク負荷表から各々のタスクに対応した負荷率5%、18%、35%と属性“fixed”, “communicative”, “somewhere”と原籍DC2、MC2, DC2、MC2, DC2、MC2とを読みだす。そして、これらのタスク名と負荷率と属性と原籍を受付用タスクキューに追加する。

【0074】(CM7)手順6720:バックアップ受

付手段6130はバックアップを依頼されたマイクロコンピュータがバックアップ後に実行すべきタスクを受付用タスクキューから選択する。具体的には、受付用タスクキューをタスクの属性による優先順位と負荷率で並べ変えて、優先順位の高いタスクのなかで限界負荷率に最も近くなる組合わせを選択し、さらにすべてを選択しても負荷率に余裕があるならば次の優先度のタスクのなかでその余裕に最も近くなる組合わせを選択する。何れかの優先順位のタスクの中で選択できないものが生じるまで、あるいは、全てのタスクが選択されるまでこの組合わせと選択を繰り返す。図10の例示では受付用タスクキューが図13に示した内容になり、タスクOS22とT21とT23とT18が最終的に選択さる。ここで、バックアップされるタスクはT18であり、バックアップされないタスクは無い、追い出されるタスクも無い。すなわち、受付用タスクキュー内の全てのタスクが選択された。

【0075】(CM8) 手順6730: バックアップ受付手段6130は前手順により選択されたタスクをスケジューラ6240に登録し、それらの実行を開始する。具体的には、バックアップされるタスクをスケジューラ6240に追加して登録し、その実行を開始させる。また追い出されるタスクの登録をスケジューラ6240から削除し、その実行を停止させる。図10の例示ではタスクT18を登録し、実行を開始させる。登録を削除するものはない。

【0076】(CM9) 手順6740: バックアップ受付手段6130はバックアップを依頼されたマイクロコンピュータの新しい実行状態を示すメッセージ6700をネットワーク1000に放送(ブロードキャスト)する。このメッセージ6700はバックアップの依頼に対する回答であることを示す情報と、バックアップ受付手順が終了したときにバックアップを依頼されたマイクロコンピュータが実行しているタスクの情報と、バックアップを依頼されたタスクの中でバックアップしたタスクを示す情報と、バックアップを依頼されたマイクロコンピュータから追い出されてその実行を停止されたタスクを示す情報を含む。このメッセージにより他のコントローラはバックアップを依頼されたマイクロコンピュータの新しい実行状態を知り、各々の記憶手段に格納している実行分担表とタスク負荷表と受付順位表を修正する。また、このメッセージ6700によりバックアップの依頼元であるバックアップ依頼手段6120は待ち状態から抜け出し、バックアップ依頼手順を再開する。図10の例示において、メッセージ6700は新しい実行状況としてマイクロコンピュータ712が実行しているタスクOS22, T21, T22, T18を示す。バックアップされたタスクとしてタスクT18を示す。追い出されたタスクとしては、該当するものがないため、追い出されたものがないことを示す。

【0077】(CM10) バックアップ受付手段6230がバックアップ受付手順を終了する。

【0078】以降、メッセージ6700により処理を再開したバックアップ依頼手順6120の処理について説明する。

【0079】(CM11) 手順6642: バックアップ依頼手段6120がメッセージ6700からバックアップされたタスクを示す情報をとりだす。そして、バックアップされたタスクがあるか否かを判定する。もし、バックアップされたタスクがないならば手順6644にて全てのマイクロコンピュータにバックアップを依頼したか否かを判定する。さらにもし、すべてのマイクロコンピュータに依頼していないならば手順6646にて外部負荷順位表を参照して、今回の依頼先に次いで負荷が小さいマイクロコンピュータを依頼先に選定する。そして、再び手順6620以降によりバックアップの依頼を繰り返す。手順6644にて、もし全てのマイクロコンピュータに依頼した後であったならば、バックアップが失敗したため、縮退運転手段を起動し縮退運転を行う。さて、手順6642に戻る。ここで、バックアップされたタスクが存在するならば手順6650へ進む。

【0080】図10の例示ではタスクT18がバックアップされたため、手順6650へ進む。

【0081】(CM12) 手順6650: バックアップ依頼手段6120はメッセージ6700の内容に応じて記憶手段6160内の実行分担表とタスク負荷表と受付順位表を変更する。図10の例示では図14の実行分担表のマイクとコンピュータDC2, MC2の実行しているタスクの欄にタスクT18を追加する。同じく負荷率の欄を58%の負荷率から84%に変更する。コントローラ2(DC2)関係の内部負荷順位を変更し、マイクロコンピュータDC2, MC1を2番に、マイクロコンピュータDC2, MC2を4番に、マイクロコンピュータDC2, MC3を1番に、マイクロコンピュータDC2, MC4を3番にする。受付順位表では受付順位1番のマイクロコンピュータDC2, MC2が負荷率84%となったためそれを後ろに回し、負荷率84%以下のマイクロコンピュータの順位を1番ずつ繰り上げる。したがって、受付順位が1番のマイクロコンピュータはDC3, MC1になる。

【0082】(CM13) 手順6660: バックアップ依頼手段6120は依頼用タスクキューからメッセージ6700に示されたところのバックアップされたタスクを削除する。また、依頼用タスクキューにメッセージ6700に示されたところの追い出されたタスクを追加する。この操作は1回のバックアップ依頼で全てのバックアップが終了しなくてもよいように設けた。すなわち、1回のバックアップでバックアップできなかったタスク、あるいは、負荷率を上げるために追い出されたタスクがある場合には、依頼用タスクキューにタスクが残

り、これを今回の依頼先と異なった依頼先に次回に依頼できるようにしたものである。図10の例示では依頼用タスクキューからタスクT18が削除され、依頼用タスクキューが空になる。

【0083】(CM14)手順6670:バックアップ依頼手段6120は依頼用タスクキューにタスクが残っているか否かを判定する。もし残っているならば再び手順6610に戻り新規に依頼先を決定し、バックアップを依頼する。また、もし、依頼用タスクキューにタスクが残っていないならば、全てのタスクのバックアップを終了したことになる。このときバックアップ依頼手段6120はバックアップ依頼手順の終了に進む。

【0084】(CM15)バックアップ依頼手段6120はバックアップ依頼手順を終了する。

【0085】以上のステップにより本実施例の分散制御システムは1台のマイクロコンピュータ(DC1, MC1)701のダウンに対して、そこで実施されていたタスクNC1, CM1, T11を自律的に分割してそれぞれをコントローラ1のマイクロコンピュータ(DC1, MC4)704, コントローラ1のマイクロコンピュータ(DC1, MC3)703, コントローラ2のマイクロコンピュータ(DC2, MC2)712においてバックアップする。

【0086】図10の例示ではダウンしたマイクロコンピュータを持つコントローラと、バックアップを依頼するコントローラが同じ場合を示した。しかし、本発明の主旨によればコントローラ1, 2, 3, 4の各々がすべてのコントローラの実行状態を記述している実行分担表を持っているために、あるコントローラのダウンにおいて他の一つのコントローラがさらに他のもう一つのコントローラに対してバックアップを依頼することが可能である。具体的には図10の例示において、コントローラ1のマイクロコンピュータ704でバックアップされたタスクCM1によって提供されるバックアップ依頼手段6120の代わりに、コントローラ2のマイクロコンピュータ711で実行されているタスクCM2によって提供されるバックアップ依頼手段6220が記憶手段6260に格納された実行分担表と受付順位表を用いてバックアップを依頼することもできる。このために、実行分担表とタスク負荷表と受付順位表は全てのコントローラ、あるいは、全てのマイクロコンピュータの状態を記憶している。図10の例示ではダウンを検出したコントローラのバックアップ依頼手段が作動するものとした。

【0087】本実施例はコントローラが4台のシステムであるが、本発明の主旨を逸脱しない範囲で2台以上のコントローラを持つシステムに拡張できる。さらに、本実施例では1台のコントローラが4個のマイクロコンピュータを含むものとしたが本発明の主旨を逸脱しない範囲で1個以上のプロセッサを持つシステムに拡張できる。

【0088】図16にバックアップ依頼元のコントローラとバックアップを依頼されたコントローラによる実行分担表とタスク負荷表と受付順位表の用い方をまとめる。自己の故障を検出したコントローラは図2の内部バックアップ手順を起動し、実行分担表を用いて自己のマイクロコンピュータの中で最も負荷の軽いマイクロコンピュータからバックアップを実行する。このとき実行すべきタスクを選定するためにタスク負荷表を用いる。当該コントローラ内の全てのマイクロコンピュータによるバックアップを実施した後もバックアップすべきタスクがある場合には当該コントローラは図3のバックアップ依頼手順を実行する。バックアップ依頼手順において当該コントローラは受付順位表を用いて最も負荷の軽いマイクロコンピュータを持つコントローラから順にバックアップを依頼してゆく。バックアップを依頼されたコントローラは指定されたマイクロコンピュータに実行させるタスクを選定するためにタスク負荷表を用いる。選定したタスクを実行し、残りのタスクを依頼元に通報する。依頼元のコントローラはバックアップすべきタスクがあるかぎり、受付順位表を用いて負荷の軽いものの選定を繰り返し、バックアップに依頼を繰り返してゆく。

【0089】さて、上記のバックアップ依頼元は故障したマイクロコンピュータを持つものであったが、当該コントローラが故障によりバックアップ手順を実行できないことも考えられる。この場合には他のコントローラがその故障を検出するまでバックアップ手順は実施されない。そこで各コントローラに適当な間隔で他のコントローラの健全性を検査させるものとする。これにより他のコントローラの故障を検出したコントローラはバックアップ依頼手順の手順6605よりバックアップ手順を開始する。この手順6605は故障したコントローラにバス70aを開放させ、そのコントローラが実行していたタスクを実行分担表から読みだし依頼用タスクキューに登録する。その後には自己の故障を検出した上記の場合と同様である。

【0090】図17及び図18は本発明の他の実施例であるところの自動車200の制御を示したものである。図17は自動車200の制御の対象を説明したものである。このように数多くの対象を一つのコントローラで制御することは、人命に係わる自動車では望ましくない。コントローラの故障が直ちにシステムの暴走、即ち、運転者による制御ができなくなり、交通事故などを引き起こす可能性が高いからである。また、色々なところにある制御対象の状態をコントローラに伝える多数の信号線あるいは各制御対象に動作を指示する多数の信号線が一つのコントローラにあつまると、信号線の総延長が大変長くなる。このため信号線の総重量が重くなり、燃費向上と排気の清浄化に必要な自動車の軽量化を阻害する。そこで、複数のコントローラを各制御対象の近傍に分散して配置し、各コントローラは各々の対象を制御す



ることが考えられる。このような分散配置型の制御システムによれば複数のコントローラの一つが故障しても、自動車全体の制御が失われることがなく、当該対象の制御が失われるにとどまる。また、多くの信号線は制御対象とその近傍のコントローラとの間を結べばよいため、信号線の総重量が軽くなる。

【0091】しかしながら、エンジンの制御やブレーキの制御さらにはパワーステアリングなどの操舵系の制御などが失われることは望ましくない。信頼性の問題を解決するために重要なコントローラに対し代替用コントローラを備えることはコスト的に難しい。そして、設置場所の問題もある。自動車においては運転者や同乗者あるいは荷物に優先的に空間を割り当てるため、コントローラに割り当てられる場所は極めて狭いからである。代替用コントローラを設けずに信頼性を向上する必要がある。さらにまた、各制御対象の制御には他の制御対象の状態量を必要とすることが多い。例えば、自動車の速度はエンジンの制御とブレーキの制御と操舵系の制御と速度そのものの表示などに使用されている。よって、信号線の総重量は未だに重く、問題である。

【0092】このような問題点を解決するためには、稼働中のコントローラが故障コントローラの負荷を自律的に代替する本発明の分散制御システムが好適である。図18はこの制御システムの構成を示したものである。まず初めに各コントローラの司る制御について説明し、その後ネットワーク1000を介した負荷の代替について説明する。

【0093】コントローラ1はエンジン1aを制御する。このコントローラ1はシリンダに流入する空気量と、エンジンの回転数と回転角度と、アクセルペダルが踏み込まれた角度と、エンジンの冷却水の温度と、排気浄化用の触媒の温度とを検出する。これらに応じて、シリンダ内の燃料と空気の比が所望の値となるように燃料噴射装置から噴射される燃料の量を操作する。具体的には燃料に掛ける圧力を制御して噴射される量を制御する。または燃料の噴射時間を制御して噴射される量を制御する。または噴射装置の噴射穴の面積を制御して噴射される量を制御する。噴射される燃料の量を制御するためにこれらの制御を組合わせて実施してもよい。また、何れか一つの制御を用いてもよい。そして、上記の状態量に応じて点火時期を操作する。これらの操作はエンジンの効率を高め出力を増加する。同時に、排気中の有害物質を低減する。また、このコントローラ1はエンジンの冷却水の温度と排気浄化用の触媒の温度が適温より低い場合にはそれらを素早く暖めるために点火時期を進角する。コントローラ1は前照燈の点灯と冷房装置の運転をコントローラ8がネットワーク1000を介して通知しとき、エンジンの停止や車速の変動を防ぐためにスロットルをやや開いてエンジンの出力を増加させる。コントローラ2がネットワーク1000を介して通知する車

速が特定の値を超過した場合には、コントローラ1は燃料の噴射量を減じて車速が特定の値を大きく越えないようにする。コントローラ2が変速中であるとネットワーク1000を介して通知したとき、このコントローラ1は変速の前後で加速度が滑らかに変化するようにエンジンの出力を調整する。コントローラ1はアクセルペダル踏み込み角度が急激に大きくなった場合には運転車が急激な加速を要求していると判断する。そして、このコントローラ1は冷房装置で消費されているエンジンの出力を車の加速に使用するためにネットワーク1000を介してコントローラ8に対して冷房装置の停止を要求する。同時に、コントローラ2にシフトダウンを要求する。

【0094】コントローラ2は変速機2aを制御する。このコントローラ2は車速を検出し、変速の基準となる車速と比較し、この結果に応じてギヤ比を変更する。具体的にはシフトアップ、シフトダウンを実施する。この時、ネットワーク1000を介してコントローラ1が通知するスロットル開度とエンジンの回転速度とコントローラ3が通知する車体姿勢に応じて、変速の基準となる車速を変更する。例えば、スロットル開度が小さくエンジンの回転速度が高いときは吸気抵抗による出力の損失が増大するため、変速の基準となる車速を下げ低速でシフトアップしエンジン回転速度を低下させる。車体姿勢が後傾しているときは坂道を昇っていると判定しシフトアップの基準車速を高め、低速ギヤを使用するようにして登坂力を維持する。

【0095】コントローラ3はサスペンション3aを制御する。このコントローラ3はサスペンションの変位と車体姿勢とを検出し、ネットワーク1000を介してコントローラ2から車速をコントローラ5から舵角を受信し、これらに応じてサスペンションの空気バネ内の気圧を制御して所望の車体姿勢にする。このコントローラ3は車体姿勢をネットワーク1000を介して他のコントローラに通知する。

【0096】コントローラ4はブレーキ4aを制御する。このコントローラ4はタイヤの回転速度を検出し、ネットワーク1000を介してコントローラ2から車速をコントローラ5から舵角を受信し、これらに応じて地面に対するタイヤの滑り率を求め、この値が所定の値以下になるように制動力を定める。さらに、旋回時に内輪に弱い制動を掛けて滑らかに旋回する。

【0097】コントローラ5はパワーステアリング5aを制御する。このコントローラ5はステアリングシャフトのねじれ角度を検出し、その角度が小さくなるようにパワーステアリングのアシストトルクを調節する。さらに、このコントローラ5は舵角を検出し、他のコントローラにネットワーク1000を介して通知する。

【0098】コントローラ6は故障を診断する。このコントローラ6はネットワーク1000を介して他のコントロ

ーラから通知される状態量や操作量を自己のシミュレーション結果と比較することによって、他のコントローラあるいは制御対象機器の故障あるいは劣化を検出する。故障あるいは劣化を検出した場合は、コントローラ8に対して故障診断結果を通知する。他のコントローラの故障を検出した場合には、前記図2のバックアップ依頼手順を開始する。

【0099】コントローラ7はネットワーク1000を介して通知された車の動作の履歴を衝撃及び熱に強い記憶装置7aに格納するドライブレコーダである。

【0100】コントローラ8はスイッチ8aとメータ8bとディスプレイ8cなどのマンマシンインターフェースを制御する。コントローラ8は運転者がスイッチ8aを操作した場合にそのオン／オフに応じて対応する装置を作動あるいは停止させる。また、ネットワーク1000を介して通知された車速やエンジンの回転速度等の状態量と故障診断の結果などをメータ8bあるいはディスプレイ8cに表示する。また、ネットワーク1000を介した他のコントローラからの依頼あるいは要求に応じて各種装置をオン／オフする。

【0101】このように各コントローラはネットワーク1000を介して検出した状態量と制御出力あるいは操作依頼を互いに通知し、そして、各コントローラは通知された状態量と制御出力と操作依頼に応じて所轄の機器を制御する。コントローラ間の状態量の授受を信号線数が最小2本であるネットワーク1000で実施することにより、先に述べた信号線の重量の問題を解決する。

【0102】つづいて、ネットワーク1000を介した自律的なバックアップについて説明する。各コントローラが自己の故障あるいは他のコントローラの故障を検出すると図2、図3、図4に示したバックアップ手順に従い故障したコントローラの負荷のバックアップを開始する。詳細な動作は前述のとおりであるのでここでは説明を省略する。このバックアップを完了するためには各コントローラの余力の合計がバックアップされる負荷より大きいことが必要である。一般に複数の異なった種類の複数の対象を複数のコントローラで制御する分散制御システムでは、各コントローラが最大負荷になるシステムの状態が異なっている。このため、複数のコントローラ全体ではバックアップに必要な余力があると考えられる。

【0103】自動車200においてこの状況を述べる。各コントローラの負荷は自動車200の運転状況に応じて変動する。例えば、エンジン1aを制御するコントローラ1はエンジン1aが高速に回転している時に負荷率が高くなり、中速の時に負荷率が小さくなる。おおむね、エンジン1aの回転速度に比例しているが、最も低速回転であるアイドリング時にはエンジンを停止させないための処理と排気の浄化のための処理を実施させるため負荷率が高まる。また、変速機2aを制御するコント

ローラ2の負荷率は車速が大きく変化しているときに高まる。特に、加速時には変速の基準速度を計算する回数を増すために負荷率が高い。逆に、定常走行しているときには低下する。さらにまた、ブレーキ4aを制御するコントローラ4はブレーキペダルが踏み込まれているときのみタイヤをスリップさせないための処理を実行させるため、このとき以外の負荷率は極めて低い。このように制御対象によって各コントローラの負荷率が高まる自動車200の走行状態が異なるため、全てのコントローラのうち何れかは負荷率が低く他のコントローラの負荷をバックアップ可能なものがある。いいくわえれば、特定のコントローラの負荷率が常に低いわけではなく、自動車200の運転状況に応じて負荷率の低いコントローラが異なる。このような自動車200の制御システムでは、故障したコントローラの負荷を前記図2、図3、図4に示したバックアップ手順により、バックアップするコントローラを負荷率の状況に応じて選定させるものである。

【0104】

【発明の効果】さらにまた、本発明により分散制御システムのコントローラの多重ダウンに対しても負荷を自律的に稼働中のコントローラに分散しバックアップすることが可能である。このため、信頼性及び耐故障性を高めることが可能である。

【図面の簡単な説明】

【図1】自律分散制御システムの論理的構成図。

【図2】内部バックアップ手順図。

【図3】自律的なバックアップ依頼手順図。

【図4】自律的なバックアップ受付手順図。

【図5】自律分散制御システムの物理的構成図。

【図6】自律分散制御コントローラ用マイクロコンピュータの構成図。

【図7】実行分担表を示す例図。

【図8】タスク負荷表を示す例図。

【図9】受付順位表を示す例図。

【図10】自律分散制御システムのバックアップ方法を示す例図。

【図11】1回目の内部バックアップにおける受付用タスクキューの例図。

【図12】2回目の内部バックアップにおける受付用タスクキューの例図。

【図13】1回目の外部バックアップにおける受付用タスクキューの例図。

【図14】内部バックアップ後の実行分担表を示す例図。

【図15】内部バックアップ後の受付順位表を示す例図。

【図16】実行分担表、タスク負荷表、受付順位表の使用方法を示す例図。

【図17】自動車制御装置を示す図。

【図18】自動車制御装置を示す図。

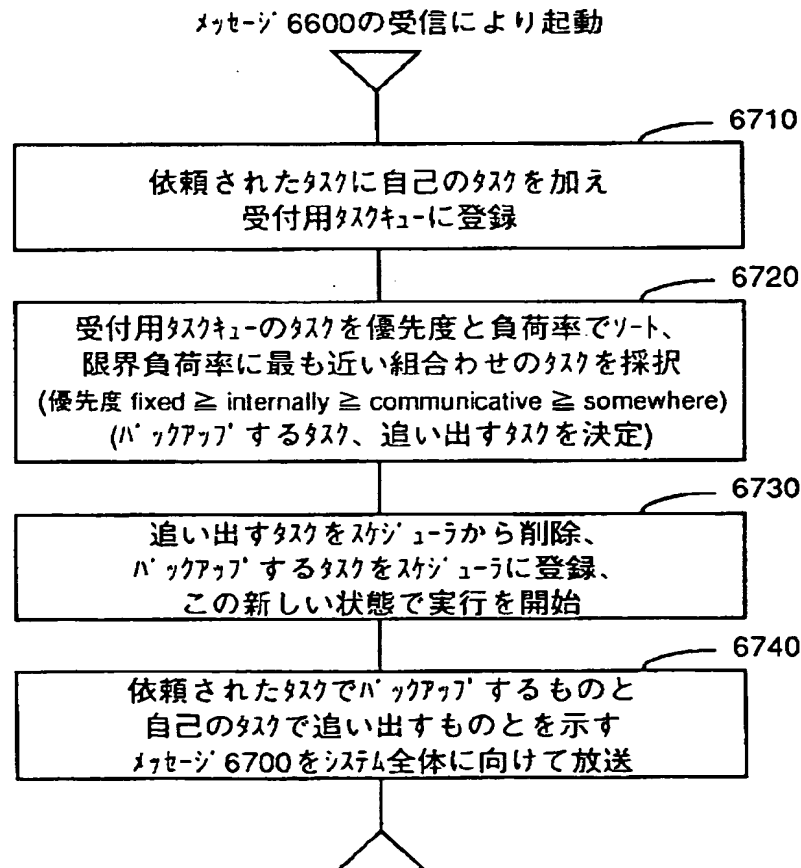
## 【符号の説明】

1, 2, 3, 4, 5, 6, 7, 8…コントローラ、1a…エンジン、2a…変速機、3a…サスペンション、4a…ブレーキ、5a…パワーステアリング、7a…記憶装置、8a…スイッチ、8b…メータ、8c…ディスプレイ、41a, 41b, 41c, 41d…ネットワーク送受信部（コネクタ）、70a, 70b, 70c, 70d…バス、71a, 71b, 71c, 71d…シリアルデータ通信用のバス、72a, 72b, 72c, 72d…故障通知用のバス、82a, 82b, 82c, 82d…グローバルメモリ、83a, 83b, 83c, 83d…入出力処理装置（I/O）、100…プラント、200…自動車、701, 702, 703, 704…コントローラ1のマイクロコンピュータ、711, 712, 713, 714…コントローラ2のマイクロコンピュータ、721, 722, 723, 724…コントローラ3のマイクロコンピュータ、731, 732, 733, 734

…コントローラ4のマイクロコンピュータ、1000…ネットワーク、6110, 6210, 6310, 6410…通信手段、6120, 6220, 6320, 6420…バックアップ依頼手段、6130, 6230, 6330, 6430…バックアップ受付手段、6140, 6240, 6340, 6440…スケジューラ、6150, 6250, 6350, 6450…制御手段、6160, 6260, 6360, 6460…記憶手段、6170, 6270, 6370, 6470…内部バックアップ手段、6180, 6280, 6380, 6480…切り離し手段、6811…プロセッサ（PR）、6812…ニューラルエンジン（NE）、6813…シーケンスエンジン（SE）、6814…ランアダプタ（LA）、6815…フリーランタイム（FRT）、6816…ローカルメモリ（LM）、6817…バスインターフェース（BIF）、6818…ダイレクトメモリアクセス制御（DMAC）、6819…マイクロコンピュータ内のバス、6820…故障制御回路。

【図4】

図 4



【図8】

図 8

受付順位表	
順位	マイクロコンピュータ
1	DC1.MC4
2	DC2.MC2
3	DC3.MC1
4	DC2.MC3
5	DC3.MC3
6	DC2.MC1
.	以下省略
.	.
.	.

【図15】

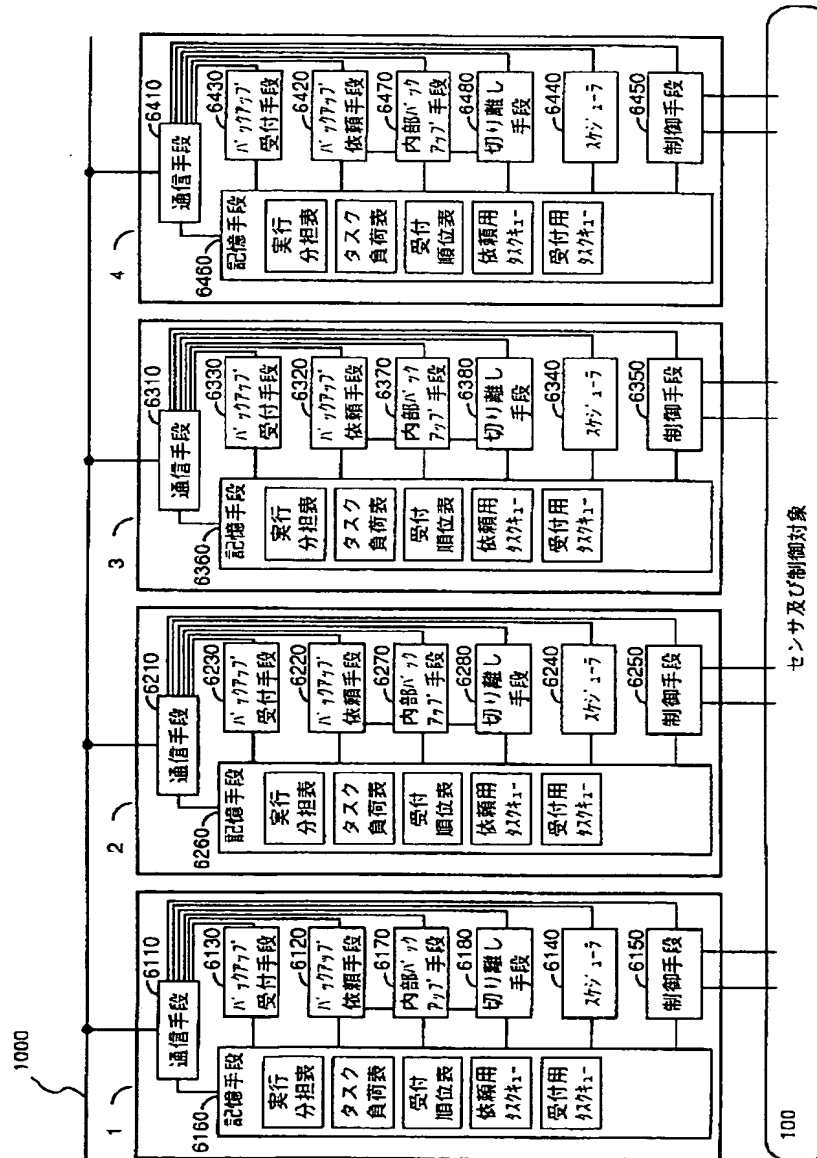
図 15

受付順位表	
順位	マイクロコンピュータ
1	DC1.MC4
2	DC2.MC2
3	DC3.MC1
4	DC2.MC3
5	DC3.MC3
6	DC2.MC1
.	以下省略
.	.
.	.

左記の様に網掛けした部分が更新されている

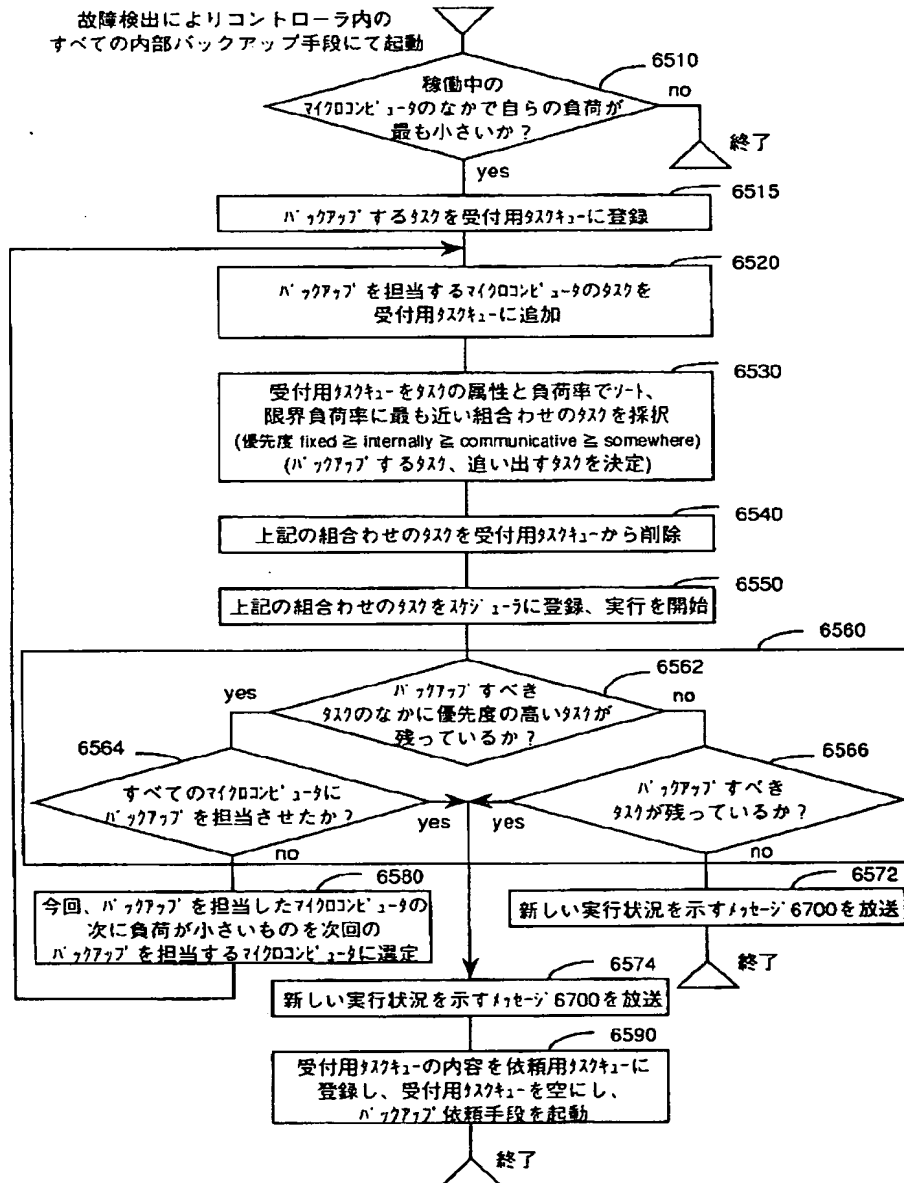
【図1】

図 1



【図 2】

図 2



【図 12】

図 12

受付用タスクキュー				
タスク	負荷率 (%)	属性	原籍	選択結果
OS13	5	fixed	DC1.MC3	選択
T11	13	communicative	DC1	選択
T16	41	communicative	DC1	選択
T18	26	somewhere	—	非選択
T15	28	somewhere	—	選択

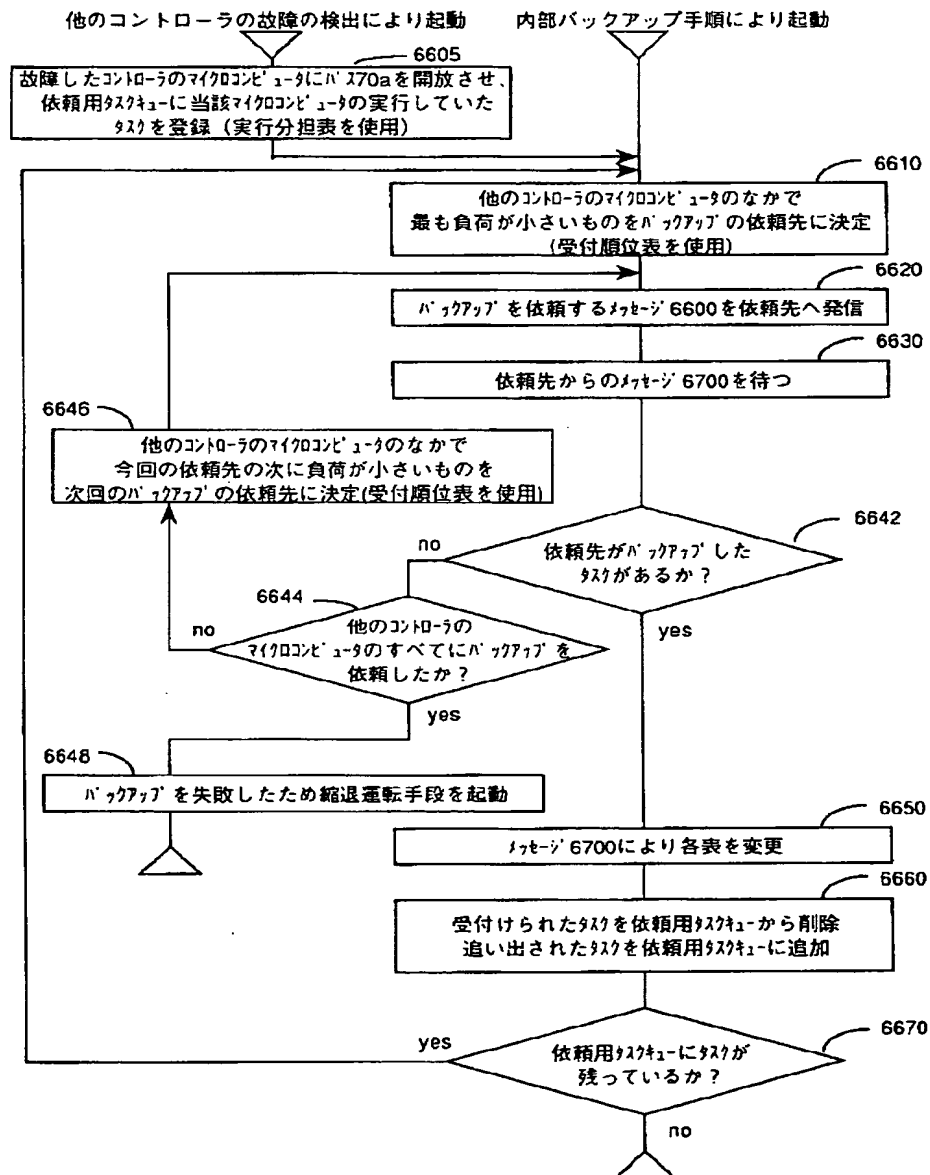
【図 13】

図 13

受付用タスクキュー				
タスク	負荷率 (%)	属性	原籍	選択結果
OS22	5	fixed	DC2.MC2	選択
T21	18	communicative	DC2	選択
T18	26	somewhere	—	選択
T22	35	somewhere	—	選択

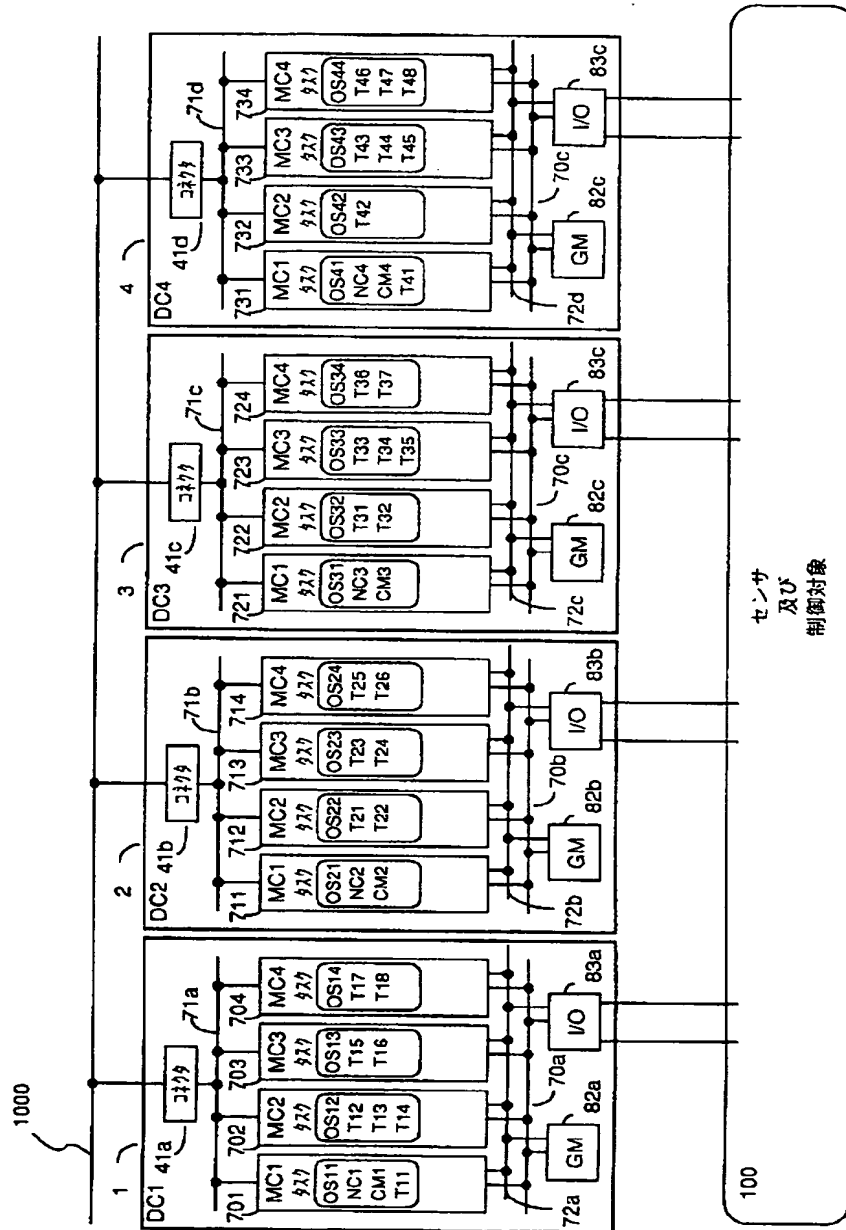
【図3】

図 3



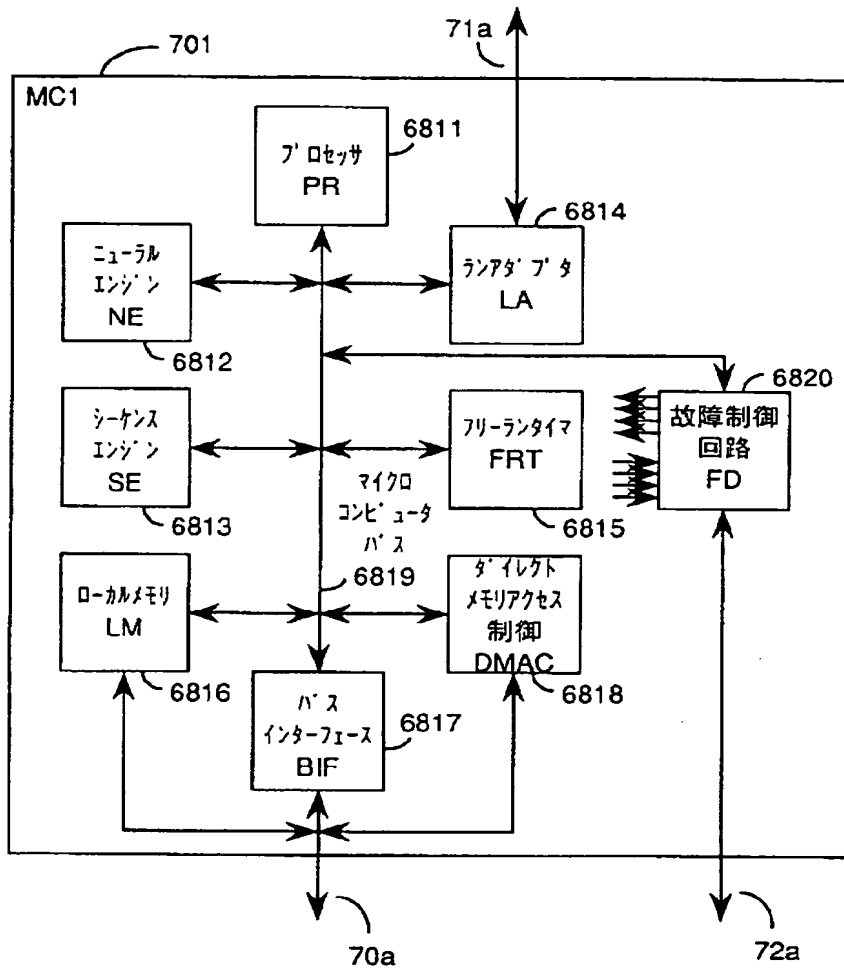
【図5】

5



【図6】

図 6



【図7】

図 7

実行分担表			
マイクロコンピュータ	タスク	負荷率 (%)	内部負荷順位
DC1.MC1	OS11, NC1, CM1, T11	72	2
DC1.MC2	OS12, T12, T13, T14	81	4
DC1.MC3	OS13, T15, T16	75	3
DC1.MC4	OS14, T17, T18	57	1
DC2.MC1	OS21, NC2, CM2	63	3
DC2.MC2	OS22, T21, T22	58	1
DC2.MC3	OS23, T23, T24	60	2
DC2.MC4	OS24, T25, T26	75	4
DC3.MC1	OS31, NC3, CM3	58	1
DC3.MC2	OS32, T31, T32	72	3
DC3.MC3	OS33, T33, T34, T35	60	2
DC3.MC4	OS34, T36, T37	74	4
	以下省略	.	.

【図11】

図 11

受付用タスクキュー				
タスク	負荷率 (%)	属性	原籍	選択結果
OS14	5	fixed	DC1.MC4	選択
NC1	26	internally	DC1	選択
CM1	28	internally	DC1	選択
T11	13	communicative	DC1	非選択
T17	25	communicative	DC1	選択
T18	26	somewhere	—	追い出し



【図9】

図9

タスク負荷表			
タスク	負荷率 (%)	属性	原籍
NC1	26	internally	DC1.MC1
CM1	28	internally	DC1.MC1
OS11	5	fixed	DC1.MC1
OS12	5	fixed	DC1.MC2
OS13	5	fixed	DC1.MC3
OS14	5	fixed	DC1.MC4
T11	13	communicative	DC1.MC1
T12	24	somewhere	DC1.MC2
T13	31	communicative	DC1.MC2
T14	21	somewhere	DC1.MC2
T15	28	somewhere	DC1.MC3
T16	41	communicative	DC1.MC3
T17	25	communicative	DC1.MC4
T18	26	somewhere	DC1.MC4
NC2	27	internally	DC2.MC1
.	.	中略	.
OS22	5	fixed	DC2.MC2
OS23	5	fixed	DC2.MC3
.	.	中略	.
T21	18	communicative	DC2.MC2
T22	35	somewhere	DC2.MC2
T23	23	communicative	DC2.MC3
T24	32	somewhere	DC2.MC3
.	.	中略	.
NC3	27	internally	DC3.MC1
CM3	26	internally	DC3.MC1
OS31	5	fixed	DC3.MC1
.	.	以下省略	.

【図14】

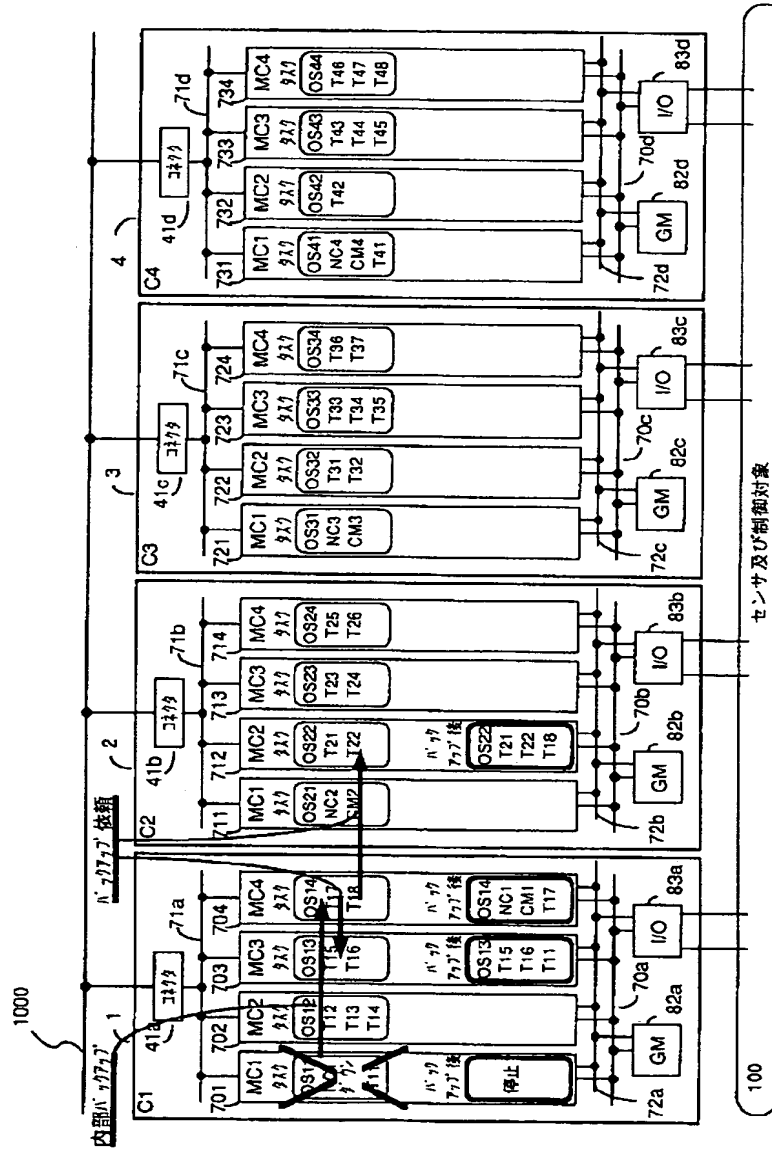
図14

実行分担表			
マイクロコンピュータ	タスク	負荷率 (%)	内部負荷順位
DC1.MC1			
DC1.MC2	OS12, T12, T13, T14	81	
DC1.MC3			
DC1.MC4			
DC2.MC1	OS21, NC2, CM2	63	3
DC2.MC2	OS22, T21, T22	58	1
DC2.MC3	OS23, T23, T24	60	2
DC2.MC4	OS24, T25, T26	75	4
DC3.MC1	OS31, NC3, CM3	58	1
DC3.MC2	OS32, T31, T32	72	3
DC3.MC3	OS33, T33, T34, T35	60	2
DC3.MC4	OS34, T36, T37	74	4
.	以下省略	.	.

左記の様に網掛けした部分が更新されている

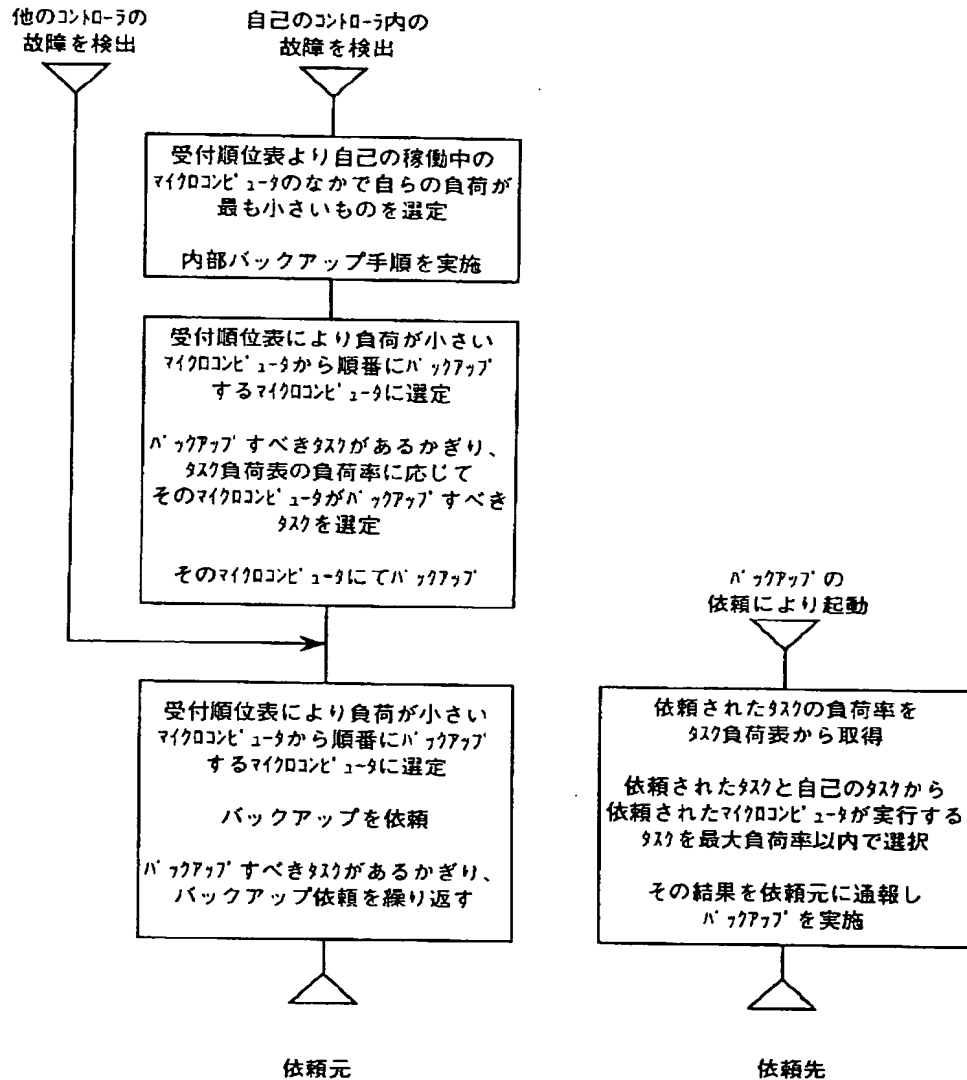
【図10】

図 10



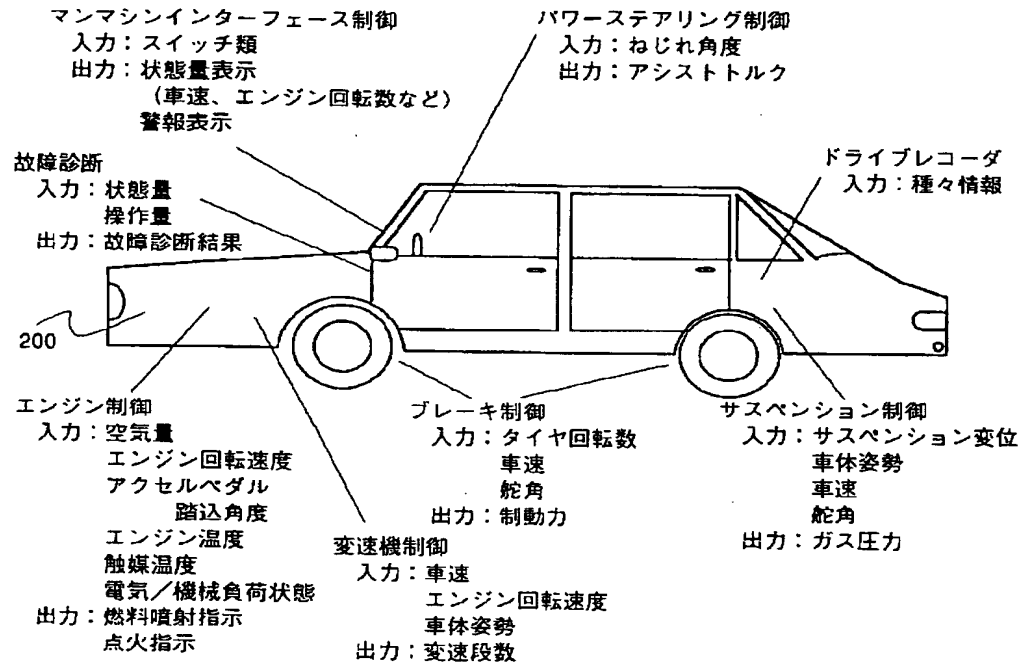
【図16】

図 16



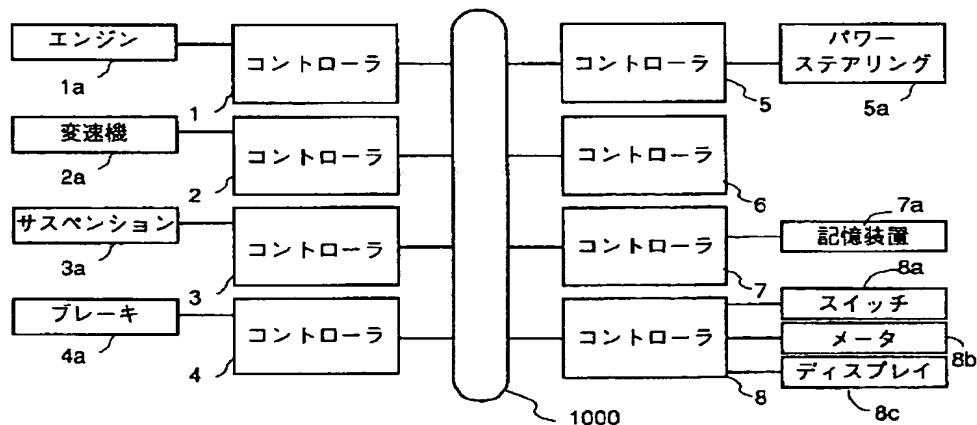
【図17】

図 17



【図18】

図 18



フロントページの続き

(51) Int. Cl.<sup>5</sup>

// G 0 5 B 9/03

識別記号

庁内整理番号

7618-3H

F I

技術表示箇所